API

UWB UWBMAC API

Qorvo

Release R12.7.0-405-gb33c5c427

Contents

1 Overview

We need an abstraction to use our uwb stack that is the same for all hardware/software architecture in order to develop cross platform user apps and the like. We divided this interface in two layers:

- UWBMAC API:

The first layer is UWBMAC API. It abstracts calls to the MAC. This layer respects the open/close principle and allows to develop any possible application with our stack.

- Helpers:

The second layer are helpers. Helpers are here to facilitate the development of application using known protocols (such as FiRa). This layer uses exclusively UWBMAC API to expose protocols related abstractions.

2 UWBMAC API

2.1 UWBMAC API

2.1.1 macro QDEPRECATED

`QDEPRECATED`(when, what)

Marks a function as deprecated.

Parameters

- `when` – when the function will be removed.
- `what` – what to use instead.

2.1.2 macro UWBMAC_MAX_CHANNEL_COUNT

UWBMAC_MAX_CHANNEL_COUNT()

Maximum number of channels in use at the same time.

2.1.3 enum uwbmac_device_state

enum **uwbmac_device_state**

State of the device.

2.1.3.1 Definition

```
enum uwbmac_device_state {
    UWBMAC_DEVICE_STATE_STOPPED,
    UWBMAC_DEVICE_STATE_STARTED,
    UWBMAC_DEVICE_STATE_BROKEN
};
```

2.1.3.2 Constants

UWBMAC_DEVICE_STATE_STOPPED
Device is stopped.

UWBMAC_DEVICE_STATE_STARTED
Device is started.

UWBMAC_DEVICE_STATE_BROKEN
Device is in a unrecoverable broken state.

2.1.4 typedef uwbmac_device_state_cb

void **uwbmac_device_state_cb**(void *user_data, enum *uwbmac_device_state* state)

Receive a device state report callback.

Parameters

- **user_data** (void*) – data given when registering this callback.
- **state** (enum *uwbmac_device_state*) – New device state.

2.1.4.1 Description

This is called when the device changes state.

2.1.4.2 Return

nothing.

2.1.5 typedef uwbmac_call_region_cb

void **uwbmac_call_region_cb**(void *user_data, uint32_t call_id, struct *uwbmac_msg* *call_params)

Receive a region call callback.

Parameters

- **user_data** (void*) – data given when registering this callback.
- **call_id** (uint32_t) – the region call identifier.
- **call_params** (struct *uwbmac_msg**) – the payload of the callback.

2.1.5.1 Return

nothing.

2.1.6 struct uwbmac_data_ops

struct **uwbmac_data_ops**

Data operations.

2.1.6.1 Definition

```
struct uwbmac_data_ops {
    void (*tx_done)(void *user_data, struct uwbmac_buf *buf, bool success);
    void (*tx_queue_stop)(void *user_data, int queue_index);
    void (*tx_queue_wake)(void *user_data, int queue_index);
    void (*rx)(void *user_data, struct uwbmac_buf *buf, int queue_index);
}
```

2.1.6.2 Members

tx_done
Called when a buffer given to *struct uwbmac_tx*() can be disposed. If NULL, buffer is released.

The success parameter is true if the transmission was done successfully.

This callback must return quickly and it must not reenter the MAC. Typical implementation will release the memory, or add the buffer in a FIFO and wake up the processing thread.

tx_queue_stop
Called to signal a queue is stopped. Application should refrain from transmitting more data frame on this queue. If NULL, ignored.

This can be called while the application is calling a MAC function.

This callback must return quickly and it must not reenter the MAC. Typical implementation will clear a flag.

tx_queue_wake

Called to signal a queue is woken up. Application can resume data frame transmission on this queue. If NULL, ignored.

This callback must return quickly and it must not reenter the MAC. Typical implementation will set a flag and wake up the processing thread.

rx

Called when a data frame has been received and that the receiving queue is not stopped. If NULL, buffer is released.

This callback must return quickly and it must not reenter the MAC. Typical implementation will add the buffer in a FIFO and wake up the processing thread.

2.1.6.3 Description

The same interface is used for any data transfer when at least one of the active regions implements it.

Data is sent and received as MPDU without the FCS, this means that the MAC header must be included, but not the MAC footer. The data must be included in a `struct uwbmac_buf`.

2.1.6.4 Transmission

To send a data frame, use the *`uwbmac_tx()`* function. The MAC will handle all the timing details and send the frame when possible. Once the frame has been sent, or when the MAC determined that the frame cannot be sent, the *`uwbmac_data_ops.tx_done`* callback is called so that the application can have a status of the transmission and reclaim memory.

The MAC can handle several queues. Frame ordering for a recipient inside a queue is guaranteed, but not between two different recipients or between two different queues.

A queue can be stopped or woken up. When a queue is stopped, the application is expected to refrain transmission of any other frame for the same queue. Any transmission attempt will result in a error returned by *`uwbmac_tx()`*. Queue state change is signaled by *`uwbmac_data_ops.tx_queue_stop`* and *`uwbmac_data_ops.tx_queue_wake`* callbacks. Queues start in the woken up state.

2.1.6.5 Reception

When a data frame is received by the MAC, the *`uwbmac_data_ops.rx`* callback is called. The callback must quickly handle the received frame and return. Typical implementation will add the received data in a FIFO and wake the processing thread. Once the frame data has been processed, the application must release the associated memory.

Application can also control the flow of data reception by calling the *`uwbmac_rx_queue_stop()`* and *`uwbmac_rx_queue_wake()`* function. Queues start in the woken up state.

2.1.7 uwbmac_get_device_count

enum qerr **`uwbmac_get_device_count`**(struct uwbmac_context *context, int *count)

Get the number of uwb chips available.

Parameters

- **`context`** (`struct uwbmac_context*`) – UWB MAC context.
- **`count`** (`int*`) – Number of uwb devices.

2.1.7.1 Return

QERR_SUCCESS or error.

2.1.8 uwbmac_get_supported_channels

enum qerr **uwbmac_get_supported_channels**(struct uwbmac_context *context, uint16_t *channels)

Get the supported UWB channels

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **channels** (uint16_t*) – (out parameter) bitmask for supported channels. First bit is for channel 0, and so on.

2.1.8.1 Return

QERR_SUCCESS or error.

2.1.9 uwbmac_init_device

enum qerr **uwbmac_init_device**(struct uwbmac_context *context, unsigned int idx)

Fill the corresponding device information.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **idx** (unsigned int) – index of the device.

2.1.9.1 NOTE

use *struct uwbmac_get_device_count* to check how many devices are present.

2.1.9.2 Return

QERR_SUCCESS or error.

2.1.10 uwbmac_register_device_state_callback

void **uwbmac_register_device_state_callback**(struct uwbmac_context *context, *uwbmac_device_state_cb* cb, void *user_data)

Register a callback for device state change.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **cb** (*uwbmac_device_state_cb*) – Callback to call on device state change.
- **user_data** (void*) – Context to give back to callback.

2.1.11 uwbmac_channel_create

enum qerr **uwbmac_channel_create**(struct uwbmac_context *context, struct uwbmac_channel *channel)

Create a new channel.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **channel** (struct uwbmac_channel*) – The channel to be created.

2.1.11.1 Return

QERR_SUCCESS or error.

2.1.12 uwbmac_channel_release

enum qerr **uwbmac_channel_release**(struct uwbmac_channel *channel)

Release a channel.

Parameters

- **channel** (struct uwbmac_channel*) – The channel to be released.

2.1.12.1 Return

QERR_SUCCESS or error.

2.1.13 uwbmac_channel_set_timeout

enum qerr **uwbmac_channel_set_timeout**(struct uwbmac_channel *channel, int timeout)

Set a timeout on a channel.

Parameters

- **channel** (struct uwbmac_channel*) – The channel .
- **timeout** (int) – The timeout in seconds.

2.1.13.1 Return

QERR_SUCCESS or error.

2.1.14 uwbmac_channel_receive

enum qerr **uwbmac_channel_receive**(struct uwbmac_channel *channel)

Ask channel to process incoming messages if any.

Parameters

- **channel** (struct uwbmac_channel*) – The channel that should process the messages.

2.1.14.1 Return

QERR_SUCCESS or error.

2.1.15 uwbmac_register_report_callback

enum qerr **uwbmac_register_report_callback**(struct uwbmac_channel *channel, *uwbmac_call_region_cb* msg_cb, void *user_data)

Register a region callback for a specific channel.

Parameters

- **channel** (`struct uwbmac_channel*`) – The channel associated with this callback.
- **msg_cb** (*`uwbmac_call_region_cb`*) – Callback to call when a report is available on this channel.
- **user_data** (`void*`) – Context to give back to callback.

2.1.15.1 Description

This function registers the callback to call in case of a mac event.

2.1.15.2 NOTE

In embedded application, the callback might be called from MAC context, large treatments should be deferred.

2.1.15.3 Return

QERR_SUCCESS or error.

2.1.16 uwbmac_register_data_ops

void **uwbmac_register_data_ops**(struct uwbmac_context *context, void *user_data, const struct *uwbmac_data_ops* *ops)

Set callbacks used for data transfer.

Parameters

- **context** (`struct uwbmac_context*`) – UWB MAC context.
- **user_data** (`void*`) – Context to give back to callback.
- **ops** (`const struct` *`uwbmac_data_ops`*`*`) – Structure with the data callbacks must be kept valid. NULL to clear callbacks.

2.1.16.1 Description

Please see *struct uwbmac_data_ops* for details.

2.1.17 uwbmac_init

enum qerr **uwbmac_init**(struct uwbmac_context **context)

Initialize the UWB MAC and return an UWB MAC context.

Parameters

- **context** (struct uwbmac_context**) – UWB MAC context.

2.1.17.1 NOTE

Some flavors of uwbmac have their own init method in their dedicated headers.

2.1.17.2 Return

QERR_SUCCESS or error.

2.1.18 uwbmac_exit

void **uwbmac_exit**(struct uwbmac_context *context)

Free the UWB MAC.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.

2.1.19 uwbmac_start

enum qerr **uwbmac_start**(struct uwbmac_context *context)

Start the device.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.

2.1.19.1 Return

QERR_SUCCESS or error.

2.1.20 uwbmac_stop

enum qerr **uwbmac_stop**(struct uwbmac_context *context)

Stop the device.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.

2.1.20.1 Return

QERR_SUCCESS or error.

2.1.21 uwbmac_is_started

bool **uwbmac_is_started**(struct uwbmac_context *context)

Return the state of UWB MAC.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.

2.1.21.1 Return

true if UWB MAC is started, false otherwise.

2.1.22 uwbmac_poll_events

enum qerr **uwbmac_poll_events**(struct uwbmac_context *context, uint64_t timeout_us)

Poll next event.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **timeout_us** (uint64_t) – Timeout, in micro-seconds, for the poll.

2.1.22.1 Description

This function is only available if you passed a NULL `event_loop_ops` to *uwbmac_init()*.

Passing 0 for `timeout_us` will make the call non-bloquing: existent pending event will be consume, and if there is no event the function will return instead of blocking.

Passing a value greated than 0 will make the function block until the timeout is reached when there is no pending event.

2.1.22.2 Return

QERR_SUCCESS or error.

2.1.23 uwbmac_set_frame_retries

enum qerr **`uwbmac_set_frame_retries`**(struct uwbmac_context *context, int retries)

Set number of retries.

Parameters

- **`context`** (`struct uwbmac_context*`) – UWB MAC context.
- **`retries`** (`int`) – Number of retries between 0 and 7.

2.1.23.1 Description

Set the number of tx frame retries when sending a frame with ACK.

2.1.23.2 Return

QERR_SUCCESS or error.

2.1.24 uwbmac_tx

enum qerr **`uwbmac_tx`**(struct uwbmac_context *context, struct uwbmac_buf *buf, int queue_index)

Send a data frame.

Parameters

- **`context`** (`struct uwbmac_context*`) – UWB MAC context.
- **`buf`** (`struct uwbmac_buf*`) – Frame buffer.
- **`queue_index`** (`int`) – Corresponding queue.

2.1.24.1 Description

Please see *`struct uwbmac_data_ops`* for details.

2.1.24.2 Return

QERR_SUCCESS or error.

2.1.25 uwbmac_tx_drop

void **uwbmac_tx_drop**(struct uwbmac_buf *buf)

Notifies a packet drop.

Parameters

- **buf** (struct uwbmac_buf*) – Frame buffer.

2.1.26 uwbmac_rx_queue_stop

void **uwbmac_rx_queue_stop**(struct uwbmac_context *context, int queue_index)

Stop a reception queue.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **queue_index** (int) – Corresponding queue.

2.1.26.1 Description

Please see *struct uwbmac_data_ops* for details.

2.1.27 uwbmac_rx_queue_wake

void **uwbmac_rx_queue_wake**(struct uwbmac_context *context, int queue_index)

Wake up a reception queue.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **queue_index** (int) – Corresponding queue.

2.1.27.1 Description

Please see *struct uwbmac_data_ops* for details.

2.1.28 uwbmac_set_channel

enum qerr **uwbmac_set_channel**(struct uwbmac_context *context, int channel)

Set UWB channel to use.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **channel** (int) – Uwb channel, supported channels depend on driver/hardware. deprecated

2.1.28.1 Return

QERR_SUCCESS or error.

2.1.29 uwbmac_get_channel

enum qerr **uwbmac_get_channel**(struct uwbmac_context *context, int *channel)

Get used UWB channel.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **channel** (int*) – Uwb channel, supported channels depend on driver/hardware. `deprecated`

2.1.29.1 Return

QERR_SUCCESS or error.

2.1.30 uwbmac_set_channel_preamble_code

enum qerr **uwbmac_set_channel_preamble_code**(struct uwbmac_context *context, int channel, int preamble_code)

Set UWB channel and preamble code to use.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **channel** (int) – UWB channel, supported channels depend on driver/hardware.
- **preamble_code** (int) – UWB preamble code.

2.1.30.1 Return

QERR_SUCCESS or error.

2.1.31 uwbmac_get_channel_preamble_code

enum qerr **uwbmac_get_channel_preamble_code**(struct uwbmac_context *context, int *channel, int *preamble_code)

Get currently used UWB channel and preamble code.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **channel** (int*) – UWB channel, supported channels depend on driver/hardware.
- **preamble_code** (int*) – UWB preamble code.

2.1.31.1 Return

QERR_SUCCESS or error.

2.1.32 uwbmac_calibration_transaction_start

enum qerr **`uwbmac_calibration_transaction_start`**(struct uwbmac_context *context)

Start a calibration transaction allowing to update configuration value(s).

Parameters

- **`context`** (`struct uwbmac_context*`) – UWB MAC context.

2.1.32.1 Description

Setting calibration value(s) as part of a transaction allows to optimize the Flash usage. When using a transaction, the new calibration values will be stored in the Flash only once, at transaction ended. While when not using a transaction will cause as many Flash write as the number of calibration values changed.

2.1.32.2 A calibration transaction should be performed by the following API call sequence

1. Start the transaction by calling `uwbmac_calibration_transaction_start`.
2. Update the calibration values by calling `uwbmac_set_calibration` for each of them.
3. End the transaction by calling `uwbmac_calibration_transaction_end`.

Warning: starting a transaction without ending it will prevent storing values in the Flash, thus causing a mismatch between persistent stored calibration, and the one used by the stack. Values updated will be lost at next reboot.

2.1.32.3 Return

QERR_SUCCESS or error.

2.1.33 uwbmac_calibration_transaction_end

enum qerr **`uwbmac_calibration_transaction_end`**(struct uwbmac_context *context)

End the ongoing calibration transaction.

Parameters

- **`context`** (`struct uwbmac_context*`) – UWB MAC context.

2.1.33.1 Description

That API should always be called after starting a transaction. See `uwbmac_calibration_transaction_start` for more details.

2.1.33.2 Return

QERR_SUCCESS or error.

2.1.34 uwbmac_set_calibration

enum qerr **uwbmac_set_calibration**(struct uwbmac_context *context, const char *key, void *value, size_t value_size)

Send a calibration key and its value

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **key** (const char*) – the calibration key name
- **value** (void*) – the value for the specified calibration key
- **value_size** (size_t) – the size of the calibration key's value

2.1.34.1 Note

setting calibration value(s) as part of a transaction allows to optimize the Flash usage. See `uwbmac_calibration_transaction_start` for more details.

2.1.34.2 Return

QERR_SUCCESS or error.

2.1.35 uwbmac_get_calibration

enum qerr **uwbmac_get_calibration**(struct uwbmac_context *context, const char *key, void *value, int *length, size_t max_length)

Retrieve a calibration value.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **key** (const char*) – The calibration key name.
- **value** (void*) – The output array for the specified calibration key.
- **length** (int*) – The length of the the resulting array.
- **max_length** (size_t) – Capacity of the array given.

2.1.35.1 Return

QERR_SUCCESS or error.

2.1.36 uwbmac_get_calibration_key_name

enum qerr **uwbmac_get_calibration_key_name**(struct uwbmac_context *context, uint16_t key_idx, char *key)

Get calibration key name for a specific key index.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **key_idx** (uint16_t) – Calibration key index to get name of.
- **key** (char*) – Calibration key name to fill-in.

2.1.36.1 Return

QERR_SUCCESS or error.

2.1.37 struct uwbmac_list_calibration_context

struct **uwbmac_list_calibration_context**

context for listing calibration keys

2.1.37.1 Definition

```
struct uwbmac_list_calibration_context {
    const char *const *list;
    size_t key_count;
    void (*dealloc_cb)(struct uwbmac_list_calibration_context *list_calibration_ctx);
}
```

2.1.37.2 Members

list
list of retrieved calibration keys

key_count
count of retrieved calibration keys

dealloc_cb
callback for freeing memory buffer

2.1.38 uwbmac_list_calibrations

enum qerr **uwbmac_list_calibrations**(struct uwbmac_context *context, struct *uwbmac_list_calibration_context* *list_calibration_ctx)

Retrieve the list calibration keys.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **list_calibration_ctx** (struct *uwbmac_list_calibration_context**) – Operation context.

2.1.38.1 Description

The list must be freed by client by calling list_calibration_ctx->dealloc_cb.

2.1.38.2 Return

QERR_SUCCESS or error.

2.1.39 uwbmac_reset_calibration

enum qerr **uwbmac_reset_calibration**(struct uwbmac_context *context)

Reset values for all calibration keys.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.

2.1.39.1 Return

QERR_SUCCESS or error.

2.1.40 uwbmac_set_pan_id

enum qerr **uwbmac_set_pan_id**(struct uwbmac_context *context, uint16_t pan_id)

Set pan id to use.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **pan_id** (uint16_t) – Pan id.

2.1.40.1 NOTE

HW Filtering is disabled if promiscuous mode is enabled.

2.1.40.2 Return

QERR_SUCCESS or error.

2.1.41 uwbmac_set_short_addr

enum qerr **uwbmac_set_short_addr**(struct uwbmac_context *context, uint16_t short_addr)

Set short address to use.

Parameters

- **context** (`struct uwbmac_context*`) – UWB MAC context.
- **short_addr** (`uint16_t`) – Short address.

2.1.41.1 NOTE

HW Filtering is disabled if promiscuous mode is enabled.

2.1.41.2 Return

QERR_SUCCESS or error.

2.1.42 uwbmac_set_extended_addr

enum qerr **uwbmac_set_extended_addr**(struct uwbmac_context *context, uint64_t extended_addr)

Set extended address to use.

Parameters

- **context** (`struct uwbmac_context*`) – UWB MAC context.
- **extended_addr** (`uint64_t`) – extended address.

2.1.42.1 NOTE

HW Filtering is disabled if promiscuous mode is enabled.

2.1.42.2 Return

QERR_SUCCESS or error.

2.1.43 uwbmac_set_promiscuous_mode

enum qerr **uwbmac_set_promiscuous_mode**(struct uwbmac_context *context, bool on)

Set promiscuous mode.

Parameters

- **context** (`struct uwbmac_context*`) – UWB MAC context.
- **on** (`bool`) – True to enable promiscuous mode.

2.1.43.1 Description

Control hardware filtering, if promiscuous mode is enabled, the hardware filtering is disabled.

2.1.43.2 Return

QERR_SUCCESS or error.

2.1.44 uwbmac_set_scheduler

enum qerr **uwbmac_set_scheduler**(struct uwbmac_context *context, const char *name, const struct *uwbmac_msg* *params)

Set the scheduler responsible for managing the schedule, and configure its parameters.

Parameters

- **context** (`struct uwbmac_context*`) – UWB MAC context.
- **name** (`const char*`) – Scheduler name.
- **params** (`const struct` *`uwbmac_msg`*`*`) – Scheduler paraameters.

2.1.44.1 Description

Device should not be started for the moment.

2.1.44.2 Return

QERR_SUCCESS or error.

2.1.45 uwbmac_get_scheduler

enum qerr **uwbmac_get_scheduler**(struct uwbmac_context *context, char *name, int max_length)

Get the scheduler name in use.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **name** (char*) – The buffer to fill with the scheduler name.
- **max_length** (int) – Length of provided buffer.

2.1.45.1 Return

QERR_SUCCESS or error.

2.1.46 uwbmac_close_scheduler

enum qerr **uwbmac_close_scheduler**(struct uwbmac_context *context)

Close the current scheduler and all regions.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.

2.1.46.1 Return

QERR_SUCCESS or error.

2.1.47 uwbmac_set_scheduler_parameters

enum qerr **uwbmac_set_scheduler_parameters**(struct uwbmac_context *context, const char *name, const struct *uwbmac_msg* *params)

Set the scheduler parameters.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **name** (const char*) – Scheduler name.
- **params** (const struct *uwbmac_msg**) – Scheduler parameters.

2.1.47.1 Return

QERR_SUCCESS or error.

2.1.48 uwbmac_get_scheduler_parameters

enum qerr **uwbmac_get_scheduler_parameters**(struct uwbmac_context *context, const char *name, struct *uwbmac_msg* *reply)

Get the scheduler parameters.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **name** (const char*) – Scheduler name.
- **reply** (struct *uwbmac_msg**) – Message filled with the parameters.

2.1.48.1 Return

QERR_SUCCESS or error.

2.1.49 uwbmac_set_regions

enum qerr **uwbmac_set_regions**(struct uwbmac_context *context, const char *scheduler_name, uint32_t region_id, const char *region_name, const struct *uwbmac_msg* *params)

Set regions that populate the schedule.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **scheduler_name** (const char*) – Scheduler name.
- **region_id** (uint32_t) – Identifier of the region, scheduler specific.
- **region_name** (const char*) – Name of region to attach to the scheduler.
- **params** (const struct *uwbmac_msg**) – Region parameters.

2.1.49.1 Return

QERR_SUCCESS or error.

2.1.50 uwbmac_set_region_parameters

enum qerr **uwbmac_set_region_parameters**(struct uwbmac_context *context, const char *scheduler_name, uint32_t region_id, const char *region_name, const struct *uwbmac_msg* *params)

Set region parameters.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **scheduler_name** (const char*) – Scheduler name.
- **region_id** (uint32_t) – Identifier of the region, scheduler specific.
- **region_name** (const char*) – Name of region to attach to the scheduler.
- **params** (const struct *uwbmac_msg**) – Region parameters.

2.1.50.1 Return

QERR_SUCCESS or error.

2.1.51 uwbmac_get_region_parameters

enum qerr **uwbmac_get_region_parameters**(struct uwbmac_context *context, const char *scheduler_name, uint32_t region_id, const char *region_name, struct *uwbmac_msg* *reply)

Get region parameters.

Parameters

- **context** (`struct uwbmac_context*`) – UWB MAC context.
- **scheduler_name** (`const char*`) – Scheduler name.
- **region_id** (`uint32_t`) – Identifier of the region, scheduler specific.
- **region_name** (`const char*`) – Name of the region to call.
- **reply** (`struct` *uwbmac_msg*`*`) – Empty message to store parameters.

2.1.51.1 NOTE

uwbmac_call_region_free must be called on the reply when done.

2.1.51.2 Return

QERR_SUCCESS or error.

2.1.52 uwbmac_call_scheduler

enum qerr **uwbmac_call_scheduler**(struct uwbmac_context *context, const char *name, uint32_t call_id, const struct *uwbmac_msg* *params, const struct uwbmac_channel *channel)

Call scheduler specific procedure.

Parameters

- **context** (`struct uwbmac_context*`) – UWB MAC context.
- **name** (`const char*`) – Scheduler name.
- **call_id** (`uint32_t`) – Identifier of the procedure, scheduler specific.
- **params** (`const struct` *uwbmac_msg*`*`) – Scheduler call parameters.
- **channel** (`const struct uwbmac_channel*`) – Channel to get response.

2.1.52.1 Return

QERR_SUCCESS or error.

2.1.53 uwbmac_call_region

int **uwbmac_call_region**(struct uwbmac_context *context, const char *scheduler_name, uint32_t region_id, const char *region_name, uint32_t call_id, const struct *uwbmac_msg* *params, const struct uwbmac_channel *channel, struct *uwbmac_msg* *reply)

Call region specific procedure.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **scheduler_name** (const char*) – Scheduler name.
- **region_id** (uint32_t) – Identifier of the region, scheduler specific.
- **region_name** (const char*) – Name of the region to call.
- **call_id** (uint32_t) – Identifier of the procedure, region specific.
- **params** (const struct *uwbmac_msg**) – Region call parameters.
- **channel** (const struct uwbmac_channel*) – Channel to get response if reply is not NULL.
- **reply** (struct *uwbmac_msg**) – If not NULL, wait for a reply and store its payload here.

2.1.53.1 NOTE

most calls to this function do not trigger a response, so reply must only be given when a reply is expected, in which case uwbmac_call_region_free must be called on the reply when done.

2.1.53.2 Return

QERR_SUCCESS, error or a positive return code.

2.1.54 uwbmac_call_region_free

void **uwbmac_call_region_free**(struct *uwbmac_msg* *reply)

Free internal resources after uwbmac_call_region.

Parameters

- **reply** (struct *uwbmac_msg**) – The reply filled in by a call to uwbmac_call_region.

2.1.55 uwbmac_get_time_ns

enum qerr **uwbmac_get_time_ns**(struct uwbmac_context *context, uint64_t *time)

Get the current MAC time.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **time** (uint64_t*) – Pointer to store current MAC time.

2.1.55.1 Return

QERR_SUCCESS or error.

2.1.56 uwbmac_get_version

const char ***uwbmac_get_version**(void)

Get the uwbmac release version.

Parameters

- **void** – no arguments

2.1.56.1 Return

The release version string.

2.1.57 struct uwbmac_pids_info

struct **uwbmac_pids_info**

UWB SPI pids.

2.1.57.1 Definition

```
struct uwbmac_pids_info {
    int spi;
    int dw3000_spi;
}
```

2.1.57.2 Members

spi
pid of dw3000 spi controller

dw3000_spi
pid of dw3000

2.1.58 uwbmac_get_spi_pids

enum qerr **uwbmac_get_spi_pids**(struct uwbmac_context *context, struct *uwbmac_pids_info* *pids)

Return spi PIDs.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **pids** (struct *uwbmac_pids_info**) – spi PIDs returned.

2.1.58.1 Return

QERR_SUCCESS or error.

2.1.59 uwbmac_set_scanning_mode

enum qerr **uwbmac_set_scanning_mode**(struct uwbmac_context *context, bool enabled)

Enable or disable scanning.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **enabled** (bool) – True to enable ieee 802.15.4 scanning.

2.1.59.1 Description

This mode is only used for IEEE 802.15.4 scanning, actual control must be handled by the MLME running on the client side.

2.1.59.2 Return

QERR_SUCCESS or error.

2.1.60 typedef uwbmac_testmode_cb_t

void **uwbmac_testmode_cb_t**(void *user_data, void *data, int length)

Receive a testmode call response.

Parameters

- **user_data** (void*) – data given when registering this callback.
- **data** (void*) – response given.
- **length** (int) – length of data.

2.1.60.1 Return

nothing.

2.1.61 uwbmac_register_testmode_callback

enum qerr **uwbmac_register_testmode_callback**(struct uwbmac_context *context, *uwbmac_testmode_cb_t* msg_cb, void *user_data)

Register a testmode callback.

Parameters

- **context** (`struct uwbmac_context*`) – UWB MAC context.
- **msg_cb** (*uwbmac_testmode_cb_t*) – Callback to call when the result of the test is available.
- **user_data** (`void*`) – Context to give back to callback.

2.1.61.1 Description

This function registers the callback to call in case of a mac event. The callback is called from MAC context, big treatments should be deferred.

2.1.61.2 NOTE

The msg sent to the callback should be freed by the APP using uwbmac_buf_free.

2.1.61.3 Return

QERR_SUCCESS or error.

2.1.62 uwbmac_call_testmode

enum qerr **uwbmac_call_testmode**(struct uwbmac_context *context, void *data, int length)

Call a test mode function.

Parameters

- **context** (`struct uwbmac_context*`) – UWB MAC context.
- **data** (`void*`) – Test data.
- **length** (`int`) – Size of test data.

2.1.62.1 Description

Test mode allows to directly call the driver. This is expected to be called for tests. Test mode may be disabled in a device.

2.1.62.2 Return

QERR_SUCCESS or error.

2.1.63 uwbmac_trace_init

enum qerr **uwbmac_trace_init**(void)

Initialize trace management module.

Parameters

- **void** – no arguments

2.1.63.1 Description

This API **must** be called by the user to initialize tracing.

2.1.63.2 NOTE

That API is only required for embedded systems.

2.1.63.3 Return

QERR_SUCCESS or error.

2.1.64 enum uwbmac_trace_module_ids

enum **uwbmac_trace_module_ids**

Unique ID for each trace module

2.1.64.1 Definition

```
enum uwbmac_trace_module_ids {
    UWBMAC_TRACE_MODULE_ID_MAIN,
    UWBMAC_TRACE_MODULE_ID_FBS,
    UWBMAC_TRACE_MODULE_ID_FIRA,
    UWBMAC_TRACE_MODULE_ID_LLD_COMMON,
    UWBMAC_TRACE_MODULE_ID_LLDD,
    UWBMAC_TRACE_MODULE_ID_LLDC,
    UWBMAC_TRACE_MODULE_ID_PCTT,
    UWBMAC_TRACE_MODULE_ID_RADAR,
    UWBMAC_TRACE_MODULE_ID_CCC,
    UWBMAC_TRACE_MODULE_NUMBER
};
```

2.1.64.2 Constants

UWBMAC_TRACE_MODULE_ID_MAIN
Main module.

UWBMAC_TRACE_MODULE_ID_FBS
FBS module.

UWBMAC_TRACE_MODULE_ID_FIRA
Fira module.

UWBMAC_TRACE_MODULE_ID_LLD_COMMON
LLD Common module.

UWBMAC_TRACE_MODULE_ID_LLDD
LLDD module.

UWBMAC_TRACE_MODULE_ID_LLDC
LLDC module.

UWBMAC_TRACE_MODULE_ID_PCTT
PCTT module.

UWBMAC_TRACE_MODULE_ID_RADAR
Radar module.

UWBMAC_TRACE_MODULE_ID_CCC
CCC module.

UWBMAC_TRACE_MODULE_NUMBER
Number of modules.

2.1.65 struct uwbmac_trace_info

struct **uwbmac_trace_info**
Trace module information

2.1.65.1 Definition

```
struct uwbmac_trace_info {
    char name[UWBMAC_TRACE_MODULE_NAME_MAX_SIZE];
    bool enable;
}
```

2.1.65.2 Members

name
name of the trace module

enable
true is trace module enabled, false otherwise

2.1.66 uwbmac_trace_module_enable

enum qerr **uwbmac_trace_module_enable**(const char *module_name, bool enable)

Enable/disable trace for a specific module

Parameters

- **module_name** (const char*) – Name of the module to set trace of.
- **enable** (bool) – true to enable, false to disable.

2.1.66.1 Description

When the user wants to enable/disable a trace module, it sets true or false the **enable** parameter.

2.1.66.2 NOTE

That API is only required for embedded systems.

2.1.66.3 Return

QERR_SUCCESS or error.

2.1.67 uwbmac_trace_module_enable_by_id

enum qerr **uwbmac_trace_module_enable_by_id**(enum *uwbmac_trace_module_ids* module_id, bool enable)

Enable/disable trace for a specific module

Parameters

- **module_id** (enum *uwbmac_trace_module_ids*) – ID of the module to set trace of.
- **enable** (bool) – true to enable, false to disable.

2.1.67.1 Description

When the user wants to enable/disable a trace module, it sets true or false the **enable** parameter.

2.1.67.2 NOTE

That API is only required for embedded systems.

2.1.67.3 Return

QERR_SUCCESS or error.

2.1.68 uwbmac_get_trace_modules

enum qerr **uwbmac_get_trace_modules**(struct *uwbmac_trace_info* **info, int *nb_modules)

Retrieve info of all trace modules available

Parameters

- **info** (struct *uwbmac_trace_info***) – output param where trace module informations are stored.
- **nb_modules** (int*) – output param where number of modules is stored.

2.1.68.1 NOTE

That API is only required for embedded systems.

2.1.68.2 Return

QERR_SUCCESS or error.

2.1.69 uwbmac_is_trace_module_enabled

bool **uwbmac_is_trace_module_enabled**(enum *uwbmac_trace_module_ids* id)

Get trace enable status for a module.

Parameters

- **id** (enum *uwbmac_trace_module_ids*) – unique ID of the module to get trace status of.

2.1.69.1 NOTE

That API is only required for embedded systems.

2.1.69.2 Return

true if enable, false if disable or ID not found.

2.1.70 struct power_state_stats

struct **power_state_stats**

Contains power statistics details for one state

2.1.70.1 Definition

```
struct power_state_stats {
    uint32_t duration_ms;
    uint32_t count;
}
```

2.1.70.2 Members

duration_ms
total duration of the state in ms

count
number of activations

2.1.71 struct uwbmac_power_stats

struct **uwbmac_power_stats**
Contains power statistics about uwb

2.1.71.1 Definition

```
struct uwbmac_power_stats {
    struct power_state_stats state_stats[UWBMAC_PWR_STATE_MAX];
    uint32_t interrupts;
}
```

2.1.71.2 Members

state_stats
statistics for each state

interrupts
number of handled interrupts

2.1.72 struct uwbmac_uwb_device_stats

struct **uwbmac_uwb_device_stats**
Contains device statistics about uwb

2.1.72.1 Definition

```
struct uwbmac_uwb_device_stats {
    int16_t temperature_hundredth_celsius;
}
```

2.1.72.2 Members

temperature_hundredth_celsius
Temperature in hundredth of degree Celsius.

2.1.73 struct uwbmac_device_info

struct **uwbmac_device_info**
Device information.

2.1.73.1 Definition

```
struct uwbmac_device_info {
    uint64_t lot_id;
    uint32_t dev_id;
    uint32_t part_id;
}
```

2.1.73.2 Members

lot_id
Lot ID.

dev_id
Device ID.

part_id
Part ID.

2.1.74 uwbmac_set_low_power_mode

enum qerr **uwbmac_set_low_power_mode**(struct uwbmac_context *context, bool enabled)
Set low power mode.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **enabled** (bool) – True to enable low power mode state.

2.1.74.1 Return

QERR_SUCCESS or error.

2.1.75 uwbmac_get_low_power_mode

bool **uwbmac_get_low_power_mode**(void)

Get low power mode S4 state.

Parameters

- **void** – no arguments

2.1.75.1 Return

True if low power mode S4 is set, otherwise false.

2.1.76 uwbmac_set_pm_min_inactivity_s4

enum qerr **uwbmac_set_pm_min_inactivity_s4**(struct uwbmac_context *context, uint32_t time_ms)

Set minimum inactivity time for S4.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **time_ms** (uint32_t) – Minimum inactivity time to get into S4, in ms.

2.1.76.1 Return

QERR_SUCCESS or error.

2.1.77 uwbmac_get_pm_min_inactivity_s4

enum qerr **uwbmac_get_pm_min_inactivity_s4**(struct uwbmac_context *context, uint32_t *time_ms)

Get minimum inactivity time for S4.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **time_ms** (uint32_t*) – minimum inactivity time to get in S4, in ms.

2.1.77.1 Return

QERR_SUCCESS or error.

2.1.78 uwbmac_se_set_key

enum qerr **uwbmac_se_set_key**(uint32_t session_id, uint8_t *key, uint8_t size, uint16_t *status)

[Not supported in QM33 SDK] Set a SE key for a given session.

Parameters

- **session_id** (uint32_t) – Id of the session.
- **key** (uint8_t*) – pointer to the session key
- **size** (uint8_t) – length of the session key, can be 128 or 256 bits.
- **status** (uint16_t*) – SE status.

2.1.78.1 Return

QERR_SUCCESS or error.

2.1.79 uwbmac_se_derive_key

enum qerr **uwbmac_se_derive_key**(const uint8_t *key, const uint8_t *data, unsigned int data_len, uint8_t *out)

[Not supported in QM33 SDK] Derive a key from a root key and derivation data.

Parameters

- **key** (const uint8_t*) – pointer to the root key
- **data** (const uint8_t*) – Derivation data.
- **data_len** (unsigned int) – Derivation data length in bytes.
- **out** (uint8_t*) – pointer to the derived key

2.1.79.1 Return

QERR_SUCCESS or error.

2.1.80 uwbmac_query_gpio_timestamp

enum qerr **uwbmac_query_gpio_timestamp**(struct uwbmac_context *context, int64_t *timestamp_us, uint8_t *sequence_number)

[Not supported in QM33 SDK] Dequeue and return gpio timestamp and sequence number.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **timestamp_us** (int64_t*) – Pointer to store the timestamp in microseconds.
- **sequence_number** (uint8_t*) – Pointer to store the sequence number.

2.1.80.1 Return

QERR_SUCCESS or error.

2.1.81 uwbmac_get_uwb_device_stats

enum qerr **uwbmac_get_uwb_device_stats**(struct uwbmac_context *context, struct *uwbmac_uwb_device_stats* *uwb_device_stats)

[Not supported in QM33 SDK] Get uwb stats.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **uwb_device_stats** (struct *uwbmac_uwb_device_stats**) – Pointer to store the uwb stats.

2.1.81.1 Return

QERR_SUCCESS or error.

2.1.82 uwbmac_reinit_crypto

enum qerr **uwbmac_reinit_crypto**(void)

Reinitialize crypto context from MCPS crypto.

Parameters

- **void** – no arguments

2.1.82.1 Return

QERR_SUCCESS or error.

2.1.83 uwbmac_get_device_info

enum qerr **uwbmac_get_device_info**(struct uwbmac_context *context, struct *uwbmac_device_info* *device_info)

Get Device Information.

Parameters

- **context** (struct uwbmac_context*) – UWB MAC context.
- **device_info** (struct *uwbmac_device_info**) – Pointer to store the device info.

2.1.83.1 Return

QERR_SUCCESS or error.

2.1.84 macro UWBMAC_BUF_CB_SIZE

UWBMAC_BUF_CB_SIZE()

Size of the control block in a network buffer structure.

2.1.85 uwbmac_buf_alloc_quota

struct uwbmac_buf ***uwbmac_buf_alloc_quota**(unsigned int size, enum mem_quota_id quota_id)

Allocate a new network buffer with requested size.

Parameters

- **size** (unsigned int) – Size of buffer.
- **quota_id** (enum mem_quota_id) – Quota to use for this allocation.

2.1.85.1 Return

Pointer to the new buffer, or NULL if no memory available.

2.1.86 uwbmac_buf_alloc

struct uwbmac_buf ***uwbmac_buf_alloc**(unsigned int size)

Allocate a new network buffer with requested size.

Parameters

- **size** (unsigned int) – Size of buffer.

2.1.86.1 Return

Pointer to the new buffer, or NULL if no memory available.

2.1.87 uwbmac_buf_free

void **uwbmac_buf_free**(struct uwbmac_buf *buf)

Release a network buffer.

Parameters

- **buf** (struct uwbmac_buf*) – Buffer to release.

2.1.88 uwbmac_buf_reserve

void **uwbmac_buf_reserve**(struct uwbmac_buf *buf, unsigned int len)

Reserve some headroom on an empty buffer.

Parameters

- **buf** (`struct uwbmac_buf*`) – Buffer where space needs to be reserved, must be empty.
- **len** (`unsigned int`) – Length to reserve.

2.1.89 uwbmac_buf_headroom

unsigned int **uwbmac_buf_headroom**(const struct uwbmac_buf *buf)

Return available space at start of buffer.

Parameters

- **buf** (`const struct uwbmac_buf*`) – Buffer.

2.1.89.1 Return

Number of allocated free bytes before the data start.

2.1.90 uwbmac_buf_tailroom

unsigned int **uwbmac_buf_tailroom**(const struct uwbmac_buf *buf)

Return available space at end of buffer.

Parameters

- **buf** (`const struct uwbmac_buf*`) – Buffer.

2.1.90.1 Return

Number of allocated free bytes after the data end.

2.1.91 uwbmac_buf_trim

void **uwbmac_buf_trim**(struct uwbmac_buf *buf, unsigned int len)

Trim data to the given length.

Parameters

- **buf** (`struct uwbmac_buf*`) – Buffer to trim.
- **len** (`unsigned int`) – New buffer length.

2.1.91.1 Description

If data is smaller than the trim length, the buffer is not modified.

2.1.91.2 NOTE

Use it only with not fragmented buffers.

2.1.92 uwbmac_buf_put

void ***uwbmac_buf_put**(struct uwbmac_buf *buf, unsigned int len)

Prepare a buffer to append new data.

Parameters

- **buf** (`struct uwbmac_buf*`) – Buffer to prepare.
- **len** (`unsigned int`) – Length of data to add.

2.1.92.1 Description

This function returns a pointer to the first byte where data must be written. The caller must make sure that there is enough space before calling this function. If fragments are used, len must not exceed the tailroom of the last fragment of the buffer.

2.1.92.2 Return

Pointer to first new byte of data.

2.1.93 uwbmac_buf_put_data

int **uwbmac_buf_put_data**(struct uwbmac_buf *buf, const void *data, unsigned int len)

Append data to a buffer.

Parameters

- **buf** (`struct uwbmac_buf*`) – Buffer to write to.
- **data** (`const void*`) – Data to append.
- **len** (`unsigned int`) – Length of new data.

2.1.93.1 Description

The caller must make sure that there is enough space before calling this function.

2.1.93.2 Return

0 or error.

2.1.94 uwbmac_buf_put_u8

void **uwbmac_buf_put_u8**(struct uwbmac_buf *buf, uint8_t data)

Append a single byte to a buffer.

Parameters

- **buf** (struct uwbmac_buf*) – Buffer to write to.
- **data** (uint8_t) – Single byte to append.

2.1.94.1 Description

The caller must make sure that there is enough space before calling this function.

2.1.95 uwbmac_buf_push

void ***uwbmac_buf_push**(struct uwbmac_buf *buf, unsigned int len)

Prepare a buffer to insert new data at buffer start.

Parameters

- **buf** (struct uwbmac_buf*) – Buffer to prepare.
- **len** (unsigned int) – Length of new data.

2.1.95.1 Description

This function returns a pointer to the first byte where data must be written. The caller must make sure that there is enough space before calling this function.

2.1.95.2 Return

Pointer to first new byte of data.

2.1.96 uwbmac_buf_pull

void **uwbmac_buf_pull**(struct uwbmac_buf *buf, unsigned int len)

Extract data from the start of buffer.

Parameters

- **buf** (struct uwbmac_buf*) – Buffer to read.
- **len** (unsigned int) – Length of data to extract.

2.1.96.1 Description

The caller must make sure that there is enough data in the buffer before calling this function.

2.1.97 uwbmac_buf_queue_init

void **uwbmac_buf_queue_init**(struct uwbmac_buf_queue *queue)

Initialize an empty queue.

Parameters

- **queue** (`struct uwbmac_buf_queue*`) – Buffer queue to initialize.

2.1.98 uwbmac_buf_queue_empty

bool **uwbmac_buf_queue_empty**(const struct uwbmac_buf_queue *queue)

Test whether a queue is empty.

Parameters

- **queue** (`const struct uwbmac_buf_queue*`) – Buffer queue.

2.1.98.1 Return

true if the queue is empty.

2.1.99 uwbmac_buf_queue_push

void **uwbmac_buf_queue_push**(struct uwbmac_buf_queue *queue, struct uwbmac_buf *buf)

Put a buffer at the start of a queue.

Parameters

- **queue** (`struct uwbmac_buf_queue*`) – Buffer queue which will receive the buffer.
- **buf** (`struct uwbmac_buf*`) – Buffer to insert.

2.1.100 uwbmac_buf_queue_put

void **uwbmac_buf_queue_put**(struct uwbmac_buf_queue *queue, struct uwbmac_buf *buf)

Put a buffer at the end of a queue.

Parameters

- **queue** (`struct uwbmac_buf_queue*`) – Buffer queue which will receive the buffer.
- **buf** (`struct uwbmac_buf*`) – Buffer to insert.

2.1.101 uwbmac_buf_queue_is_last

bool **uwbmac_buf_queue_is_last**(const struct uwbmac_buf_queue *queue, const struct uwbmac_buf *buf)

Check if buf is the last entry in the queue.

Parameters

- **queue** (`const struct uwbmac_buf_queue*`) – Queue head.
- **buf** (`const struct uwbmac_buf*`) – Current buffer.

2.1.101.1 Return

True if buf is the last buffer on the list.

2.1.102 uwbmac_buf_queue_next

struct uwbmac_buf ***uwbmac_buf_queue_next**(const struct uwbmac_buf_queue *queue, const struct uwbmac_buf *buf)

Return the next packet in the queue.

Parameters

- **queue** (`const struct uwbmac_buf_queue*`) – Queue head.
- **buf** (`const struct uwbmac_buf*`) – Current buffer.

2.1.102.1 Return

Next packet in the queue.

2.1.103 uwbmac_buf_queue_peek

struct uwbmac_buf ***uwbmac_buf_queue_peek**(struct uwbmac_buf_queue *queue)

Peek a buffer from the start of a queue.

Parameters

- **queue** (`struct uwbmac_buf_queue*`) – Buffer queue to peek the buffer from.

2.1.103.1 Description

Buffer is left in the queue.

2.1.103.2 Return

The peeked buffer, or NULL if the queue is empty.

2.1.104 uwbmac_buf_queue_pop

struct uwbmac_buf ***uwbmac_buf_queue_pop**(struct uwbmac_buf_queue *queue)

Get and remove a buffer from the start of a queue.

Parameters

- **queue** (`struct uwbmac_buf_queue*`) – Buffer queue to extract the buffer from.

2.1.104.1 Return

The extracted buffer, or NULL if the queue is empty.

2.1.105 uwbmac_buf_queue_purge

void **uwbmac_buf_queue_purge**(struct uwbmac_buf_queue *queue)

Release all buffers in a queue.

Parameters

- **queue** (`struct uwbmac_buf_queue*`) – Buffer queue to purge

2.1.106 uwbmac_buf_get_next_frag

struct uwbmac_buf ***uwbmac_buf_get_next_frag**(struct uwbmac_buf *buf)

Retrieve next fragment data.

Parameters

- **buf** (`struct uwbmac_buf*`) – Buffer.

2.1.106.1 Return

Pointer to next fragment or NULL.

2.1.107 uwbmac_buf_get_data

uint8_t ***uwbmac_buf_get_data**(struct uwbmac_buf *buf)

Retrieve pointer to buffer data

Parameters

- **buf** (`struct uwbmac_buf*`) – Buffer.

2.1.107.1 Return

Pointer to first byte of buffer data.

2.1.108 uwbmac_buf_get_len

unsigned int **uwbmac_buf_get_len**(struct uwbmac_buf *buf)

Retrieve buffer data length.

Parameters

- **buf** (struct uwbmac_buf*) – Buffer.

2.1.108.1 Return

Buffer data length.

2.1.109 uwbmac_buf_get_frag_len

unsigned int **uwbmac_buf_get_frag_len**(struct uwbmac_buf *buf)

Retrieve current fragment data length.

Parameters

- **buf** (struct uwbmac_buf*) – Buffer/Fragment.

2.1.109.1 Return

Fragment data length.

2.1.110 uwbmac_buf_get_size

unsigned int **uwbmac_buf_get_size**(struct uwbmac_buf *buf)

Retrieve buffer size.

Parameters

- **buf** (struct uwbmac_buf*) – Buffer.

2.1.110.1 Return

Buffer size.

2.1.111 uwbmac_buf_set_queue_mapping

void **uwbmac_buf_set_queue_mapping**(struct uwbmac_buf *buf, uint16_t value)

Set queue mapping field of buffer.

Parameters

- **buf** (struct uwbmac_buf*) – Buffer to write to.
- **value** (uint16_t) – queue_mapping value to set.

2.1.112 uwbmac_buf_free_msg_priv

void **uwbmac_buf_free_msg_priv**(struct *uwbmac_msg* *msg)

Free priv member of msg.

Parameters

- **msg** (struct *uwbmac_msg**) – Message to free.

2.1.113 struct uwbmac_msg

struct **uwbmac_msg**

Message container.

2.1.113.1 Definition

```
struct uwbmac_msg {
    struct uwbmac_msg *parent;
    void *payload;
    uint8_t *position;
    void *priv;
    int length;
    int size;
    bool add_failed;
}
```

2.1.113.2 Members

parent
Pointer to the parent, for nested messages.

payload
Pointer to the payload to be sent.

position
Pointer to the payload being written.

priv
Pointer to private data to keep around.

length
Length of the payload.

size
Capacity of the payload buffer.

add_failed
Set to true when an add call fails.

2.1.113.3 NOTE

do not access the fields directly, use the helper functions below.

2.1.114 uwbmac_msg_free_priv

void **uwbmac_msg_free_priv**(struct *uwbmac_msg* *msg)
Free uwbmac_msg priv member.

Parameters

- **msg** (struct *uwbmac_msg**) – Message to free.

2.1.115 uwbmac_msg_init

void **uwbmac_msg_init**(struct *uwbmac_msg* *msg, void *payload, int length, int size)
Initialize message from payload/length

Parameters

- **msg** (struct *uwbmac_msg**) – Message to initialize.
- **payload** (void*) – Pointer to the payload.
- **length** (int) – Length of the payload.
- **size** (int) – Total size available in the payload.

2.1.116 uwbmac_msg_copy

bool **uwbmac_msg_copy**(struct *uwbmac_msg* *msg, void *payload, int length)
Copy a payload into a message

Parameters

- **msg** (struct *uwbmac_msg**) – Message to initialize.
- **payload** (void*) – Pointer to the payload.
- **length** (int) – Length of the payload.

2.1.116.1 Return

true if there was enough space to do the copy, false otherwise.

2.1.117 uwbmac_msg_payload

void ***uwbmac_msg_payload**(const struct *uwbmac_msg* *msg)

Get the message payload

Parameters

- **msg** (`const struct` *uwbmac_msg**) – Message to use.

2.1.117.1 Return

the message payload or NULL is msg is NULL.

2.1.118 uwbmac_msg_length

int **uwbmac_msg_length**(const struct *uwbmac_msg* *msg)

Get the message payload length

Parameters

- **msg** (`const struct` *uwbmac_msg**) – Message to use.

2.1.118.1 Return

the message payload length.

2.1.119 uwbmac_msg_size

int **uwbmac_msg_size**(const struct *uwbmac_msg* *msg)

Get the message capacity

Parameters

- **msg** (`const struct` *uwbmac_msg**) – Message to use.

2.1.119.1 Return

the message capacity.

2.1.120 enum uwbmac_payload_type

enum **uwbmac_payload_type**

UWB MAC serializable types.

2.1.120.1 Definition

```
enum uwbmac_payload_type {
    UWBMAC_PAYLOAD_TYPE_NONE,
    UWBMAC_PAYLOAD_TYPE_FLAG,
    UWBMAC_PAYLOAD_TYPE_BOOL,
    UWBMAC_PAYLOAD_TYPE_S8,
    UWBMAC_PAYLOAD_TYPE_S16,
    UWBMAC_PAYLOAD_TYPE_S32,
    UWBMAC_PAYLOAD_TYPE_S64,
    UWBMAC_PAYLOAD_TYPE_U8,
    UWBMAC_PAYLOAD_TYPE_U16,
    UWBMAC_PAYLOAD_TYPE_U32,
    UWBMAC_PAYLOAD_TYPE_U64,
    UWBMAC_PAYLOAD_TYPE_STRING,
    UWBMAC_PAYLOAD_TYPE_BINARY,
    UWBMAC_PAYLOAD_TYPE_NESTED
};
```

2.1.120.2 Constants

UWBMAC_PAYLOAD_TYPE_NONE
No data to recover.

UWBMAC_PAYLOAD_TYPE_FLAG
Flag - no data.

UWBMAC_PAYLOAD_TYPE_BOOL
Boolean.

UWBMAC_PAYLOAD_TYPE_S8
8 bit signed integer.

UWBMAC_PAYLOAD_TYPE_S16
16 bit signed integer.

UWBMAC_PAYLOAD_TYPE_S32
32 bit signed integer.

UWBMAC_PAYLOAD_TYPE_S64
64 bit signed integer.

UWBMAC_PAYLOAD_TYPE_U8
8 bit unsigned integer.

UWBMAC_PAYLOAD_TYPE_U16
16 bit unsigned integer.

UWBMAC_PAYLOAD_TYPE_U32
32 bit unsigned integer.

UWBMAC_PAYLOAD_TYPE_U64
64 bit unsigned integer.

UWBMAC_PAYLOAD_TYPE_STRING
NULL terminated character string.

UWBMAC_PAYLOAD_TYPE_BINARY
Binary object.

UWBMAC_PAYLOAD_TYPE_NESTED
Nested payload.

2.1.121 struct uwbmac_parser_element

struct **uwbmac_parser_element**
Helper to manipulate each UWB MAC elements.

2.1.121.1 Definition

```
struct uwbmac_parser_element {
    void *data;
    uint16_t length;
    int *rlength;
    uint8_t type;
    uint8_t flags;
}
```

2.1.121.2 Members

data
Pointer to data

length
Data length max

rlength
Data length found

type
Expected data type

flags
Tag mandatory/present

2.1.122 uwbmac_parser_init_msg

void **uwbmac_parser_init_msg**(struct *uwbmac_msg* *msg, void *payload, int length)
Initialise on-stack uwbmac_msg.

Parameters

- **msg** (struct *uwbmac_msg**) – Message being initialized.
- **payload** (void*) – Payload to parse.
- **length** (int) – Length of the payload to parse.

2.1.123 uwbmac_parser_read

enum qerr **uwbmac_parser_read**(struct *uwbmac_msg* *msg, struct *uwbmac_parser_element* elements, int tag_max)

Read and parse payload.

Parameters

- **msg** (struct *uwbmac_msg**) – Message to parse.
- **elements** (struct *uwbmac_parser_element*) – Array of elements.
- **tag_max** (int) – Maximum tag value (number of elements minus one).

2.1.123.1 Return

QERR_SUCCESS or error.

2.1.124 uwbmac_parser_read_array

enum qerr **uwbmac_parser_read_array**(struct *uwbmac_msg* *msg, struct *uwbmac_parser_element* elements, int tag_max, void *entry, int n, struct uwbmac_parser_read_array_info *info, bool *keep_going)

Read and parse array payload.

Parameters

- **msg** (struct *uwbmac_msg**) – Message to parse.
- **elements** (struct *uwbmac_parser_element*) – Array of elements.
- **tag_max** (int) – Maximum tag value (number of elements minus one).
- **entry** (void*) – Array entry.
- **n** (int) – Number of array entries found so far.
- **info** (struct uwbmac_parser_read_array_info*) – Internal loop data.
- **keep_going** (bool*) – Whether there are elements left in the array.

2.1.124.1 Return

QERR_SUCCESS or error.

2.1.125 uwbmac_parser_init_nested_loop

enum qerr **uwbmac_parser_init_nested_loop**(struct *uwbmac_msg* *msg, struct uwbmac_parser_read_array_info *info)

Init internal loop data for nested iteration

Parameters

- **msg** (struct *uwbmac_msg**) – Message to parse.
- **info** (struct uwbmac_parser_read_array_info*) – Internal loop data.

2.1.125.1 Description

This call is once before looping on each element with uwbmac_parser_next_nested_loop_element.

2.1.125.2 Return

QERR_SUCCESS or error.

2.1.126 uwbmac_parser_next_nested_loop_element

enum qerr **uwbmac_parser_next_nested_loop_element**(struct *uwbmac_msg* *msg, struct *uwbmac_msg* *nested, struct uwbmac_parser_read_array_info *info, bool *keep_going)

Init nested with the next nested element

Parameters

- **msg** (struct *uwbmac_msg**) – Message to parse.
- **nested** (struct *uwbmac_msg**) – Message to init.
- **info** (struct uwbmac_parser_read_array_info*) – Internal loop data.
- **keep_going** (bool*) – Whether there are elements left in the array.

2.1.126.1 Description

If keep_going is false, you've reached the end of the nested elements and nested was not initialized.

If keep_going is true, nested is initialized to point to the next element. You will need to setup your parsing and call uwbmac_parser_read.

2.1.126.2 Return

QERR_SUCCESS or error.

2.1.127 uwbmac_parser_is_present

bool **uwbmac_parser_is_present**(struct *uwbmac_parser_element* elements, int tag)

Get tag presence status.

Parameters

- **elements** (struct *uwbmac_parser_element*) – Array of elements.
- **tag** (int) – Tag in the payload.

2.1.127.1 Return

true if present, false otherwise.

2.1.128 uwbmac_parser_add

void **uwbmac_parser_add**(struct *uwbmac_parser_element* elements, int tag, enum *uwbmac_payload_type* type, void *data, int *rlength, int length, bool mandatory)

Set elements.

Parameters

- **elements** (struct *uwbmac_parser_element*) – Array of elements.
- **tag** (int) – Tag in the payload.
- **type** (enum *uwbmac_payload_type*) – Expected tag type.
- **data** (void*) – Some pointer.
- **rlength** (int*) – Actual payload's length.
- **length** (int) – Maximum expected length in the payload.
- **mandatory** (bool) – Whether the element is mandatory in the message.

2.1.128.1 NOTE

do not call directly, use one of the provided helpers.

2.1.129 uwbmac_parser_add_none

void **uwbmac_parser_add_none**(struct *uwbmac_parser_element* elements, int tag)

Set element to receive nothing.

Parameters

- **elements** (struct *uwbmac_parser_element*) – Array of elements.
- **tag** (int) – Tag in the payload.

2.1.130 uwbmac_parser_add_flag

void **uwbmac_parser_add_flag**(struct *uwbmac_parser_element* elements, int tag, bool *data, bool mandatory)

Set element to receive an empty tag.

Parameters

- **elements** (struct *uwbmac_parser_element*) – Array of elements.
- **tag** (int) – Tag in the payload.
- **data** (bool*) – Pointer to the boolean to set if present.
- **mandatory** (bool) – Whether the element is mandatory in the message.

2.1.131 uwbmac_parser_add_bool

void **uwbmac_parser_add_bool**(struct *uwbmac_parser_element* elements, int tag, bool *data, bool mandatory)

Set element to receive a boolean.

Parameters

- **elements** (struct *uwbmac_parser_element*) – Array of elements.
- **tag** (int) – Tag in the payload.
- **data** (bool*) – Pointer to the boolean to fill in.
- **mandatory** (bool) – Whether the element is mandatory in the message.

2.1.132 uwbmac_parser_add_s8

void **uwbmac_parser_add_s8**(struct *uwbmac_parser_element* elements, int tag, int8_t *data, bool mandatory)

Set element to receive a signed 8-bit integer.

Parameters

- **elements** (struct *uwbmac_parser_element*) – Array of elements.
- **tag** (int) – Tag in the payload.
- **data** (int8_t*) – Pointer to the integer to fill in.
- **mandatory** (bool) – Whether the element is mandatory in the message.

2.1.133 uwbmac_parser_add_s16

void **uwbmac_parser_add_s16**(struct *uwbmac_parser_element* elements, int tag, int16_t *data, bool mandatory)

Set element to receive a signed 16-bit integer.

Parameters

- **elements** (struct *uwbmac_parser_element*) – Array of elements.
- **tag** (int) – Tag in the payload.
- **data** (int16_t*) – Pointer to the integer to fill in.
- **mandatory** (bool) – Whether the element is mandatory in the message.

2.1.134 uwbmac_parser_add_s32

void **uwbmac_parser_add_s32**(struct *uwbmac_parser_element* elements, int tag, int32_t *data, bool mandatory)

Set element to receive a signed 32-bit integer.

Parameters

- **elements** (struct *uwbmac_parser_element*) – Array of elements.
- **tag** (int) – Tag in the payload.
- **data** (int32_t*) – Pointer to the integer to fill in.
- **mandatory** (bool) – Whether the element is mandatory in the message.

2.1.135 uwbmac_parser_add_s64

void **uwbmac_parser_add_s64**(struct *uwbmac_parser_element* elements, int tag, int64_t *data, bool mandatory)

Set element to receive a signed 64-bit integer.

Parameters

- **elements** (struct *uwbmac_parser_element*) – Array of elements.
- **tag** (int) – Tag in the payload.
- **data** (int64_t*) – Pointer to the integer to fill in.
- **mandatory** (bool) – Whether the element is mandatory in the message.

2.1.136 uwbmac_parser_add_u8

void **uwbmac_parser_add_u8**(struct *uwbmac_parser_element* elements, int tag, uint8_t *data, bool mandatory)

Set element to receive an unsigned 8-bit integer.

Parameters

- **elements** (struct *uwbmac_parser_element*) – Array of elements.
- **tag** (int) – Tag in the payload.
- **data** (uint8_t*) – Pointer to the integer to fill in.
- **mandatory** (bool) – Whether the element is mandatory in the message.

2.1.137 uwbmac_parser_add_u16

void **uwbmac_parser_add_u16**(struct *uwbmac_parser_element* elements, int tag, uint16_t *data, bool mandatory)

Set element to receive an unsigned 16-bit integer.

Parameters

- **elements** (struct *uwbmac_parser_element*) – Array of elements.
- **tag** (int) – Tag in the payload.
- **data** (uint16_t*) – Pointer to the integer to fill in.
- **mandatory** (bool) – Whether the element is mandatory in the message.

2.1.138 uwbmac_parser_add_u32

void **uwbmac_parser_add_u32**(struct *uwbmac_parser_element* elements, int tag, uint32_t *data, bool mandatory)

Set element to receive an unsigned 32-bit integer.

Parameters

- **elements** (struct *uwbmac_parser_element*) – Array of elements.
- **tag** (int) – Tag in the payload.
- **data** (uint32_t*) – Pointer to the integer to fill in.
- **mandatory** (bool) – Whether the element is mandatory in the message.

2.1.139 uwbmac_parser_add_u64

void **uwbmac_parser_add_u64**(struct *uwbmac_parser_element* elements, int tag, uint64_t *data, bool mandatory)

Set element to receive an unsigned 64-bit integer.

Parameters

- **elements** (struct *uwbmac_parser_element*) – Array of elements.
- **tag** (int) – Tag in the payload.
- **data** (uint64_t*) – Pointer to the integer to fill in.
- **mandatory** (bool) – Whether the element is mandatory in the message.

2.1.140 uwbmac_parser_add_string

void **uwbmac_parser_add_string**(struct *uwbmac_parser_element* elements, int tag, char *data, int max_length, bool mandatory)

Set element to receive a string.

Parameters

- **elements** (struct *uwbmac_parser_element*) – Array of elements.
- **tag** (int) – Tag in the payload.
- **data** (char*) – Pointer to the string to fill in.
- **max_length** (int) – Length available, including terminating NUL character.
- **mandatory** (bool) – Whether the element is mandatory in the message.

2.1.140.1 NOTE

The payload string is copied from the payload to the provided string.

2.1.141 uwbmac_parser_add_binary

void **uwbmac_parser_add_binary**(struct *uwbmac_parser_element* elements, int tag, void *data, int *length, int max_length, bool mandatory)

Set element to receive a binary object.

Parameters

- **elements** (struct *uwbmac_parser_element*) – Array of elements.
- **tag** (int) – Tag in the payload.
- **data** (void*) – Pointer to the object to fill in.
- **length** (int*) – Actual length.
- **max_length** (int) – Length available.
- **mandatory** (bool) – Whether the element is mandatory in the message.

2.1.141.1 NOTE

The payload object is copied from the payload to the provided object.

2.1.142 uwbmac_parser_add_nested

void **uwbmac_parser_add_nested**(struct *uwbmac_parser_element* elements, int tag, struct *uwbmac_msg* *nested, bool mandatory)

Set element to receive a nested message.

Parameters

- **elements** (struct *uwbmac_parser_element*) – Array of elements.
- **tag** (int) – Tag in the payload.
- **nested** (struct *uwbmac_msg**) – Pointer to the structure to keep internal data.
- **mandatory** (bool) – Whether the element is mandatory in the message.

2.1.143 uwbmac_msg_read_tag

enum qerr **uwbmac_msg_read_tag**(struct *uwbmac_msg* *msg, int *tag, bool *is_nested, int *rem)

Tell current message element tag and data type.

Parameters

- **msg** (struct *uwbmac_msg**) – message in its current reading state
- **tag** (int*) – output value
- **is_nested** (bool*) – output value telling whether data is a nested message to binary data.
- **rem** (int*) – in/out remaining size to parse in message

2.1.143.1 Description

Set message state to the next element. Used to serialize the message without knowing the meaning of its elements. NB: *rem* parameter must be updated synchronously to msg *position* pointer.

2.1.143.2 Return

status QERR_SUCCESS or QERR_EINVAL.

2.1.144 uwbmac_msg_read_nested

enum qerr **uwbmac_msg_read_nested**(struct *uwbmac_msg* *msg, struct *uwbmac_msg* *nested, int *rem)

Get the current element as a nested msg

Parameters

- **msg** (struct *uwbmac_msg**) – message in its current reading state
- **nested** (struct *uwbmac_msg**) – messge to be initialized to point to the nested part of msg
- **rem** (int*) – in/out remaining size to parse in message

2.1.144.1 Description

Actual type not checked, *uwbmac_msg_read_tag()* should be used before. Set message state to the next element. Used to serialize the message without knowing the meaning of its elements.

NB: *rem* parameter must be updated synchronously to msg *position* pointer.

2.1.144.2 Return

status QERR_SUCCESS or QERR_EINVAL.

2.1.145 uwbmac_msg_read_data

enum qerr **uwbmac_msg_read_data**(struct *uwbmac_msg* *msg, uint8_t **data, size_t *length, int *rem)

Get the current element as binary data

Parameters

- **msg** (struct *uwbmac_msg**) – message in its current reading state
- **data** (uint8_t**) – output, set to point to the data
- **length** (size_t*) – output, set to the data length
- **rem** (int*) – in/out remaining size to parse in message

2.1.145.1 Description

Actual type not checked, *uwbmac_msg_read_tag()* should be used before. Set message state to the next element. Used to serialize the message without knowing the meaning of its elements.

NB: *rem* parameter must be updated synchronously to msg *position* pointer.

2.1.145.2 Return

status QERR_SUCCESS or QERR_EINVAL.

2.1.146 uwbmac_writer_init_msg

void **uwbmac_writer_init_msg**(struct *uwbmac_msg* *msg, void *payload, int size)

Initialise on-stack uwbmac_msg.

Parameters

- **msg** (struct *uwbmac_msg**) – Message being initialized.
- **payload** (void*) – Payload buffer to fill in.
- **size** (int) – Size of the payload buffer.

2.1.147 uwbmac_writer_success

enum qerr **uwbmac_writer_success**(const struct *uwbmac_msg* *msg)

Check that all 'add' operations succeeded.

Parameters

- **msg** (const struct *uwbmac_msg**) – Message being written.

2.1.147.1 Return

QERR_SUCCESS on success, QERR_EINVAL otherwise.

2.1.148 uwbmac_writer_add

enum qerr **uwbmac_writer_add**(struct *uwbmac_msg* *msg, int tag, const void *data, int length)

Add tag and data to the message.

Parameters

- **msg** (struct *uwbmac_msg**) – Message being written.
- **tag** (int) – Tag in the payload.
- **data** (const void*) – Payload related to tag.
- **length** (int) – Payload length.

2.1.148.1 Return

QERR_SUCCESS or error.

2.1.149 uwbmac_writer_add_flag

enum qerr **uwbmac_writer_add_flag**(struct *uwbmac_msg* *msg, int tag)

Add an empty tag to the message.

Parameters

- **msg** (struct *uwbmac_msg**) – Message being written.
- **tag** (int) – Tag in the payload.

2.1.149.1 Return

QERR_SUCCESS or error.

2.1.150 uwbmac_writer_add_bool

enum qerr **uwbmac_writer_add_bool**(struct *uwbmac_msg* *msg, int tag, bool value)

Add a boolean to the message.

Parameters

- **msg** (struct *uwbmac_msg**) – Message being written.
- **tag** (int) – Tag in the payload.
- **value** (bool) – Value to add.

2.1.150.1 Return

QERR_SUCCESS or error.

2.1.151 uwbmac_writer_add_s8

enum qerr **uwbmac_writer_add_s8**(struct *uwbmac_msg* *msg, int tag, int8_t value)

Add a signed 8-bit integer to the message.

Parameters

- **msg** (struct *uwbmac_msg**) – Message being written.
- **tag** (int) – Tag in the payload.
- **value** (int8_t) – Value to add.

2.1.151.1 Return

QERR_SUCCESS or error.

2.1.152 uwbmac_writer_add_s16

enum qerr **uwbmac_writer_add_s16**(struct *uwbmac_msg* *msg, int tag, int16_t value)

Add a signed 16-bit integer to the message.

Parameters

- **msg** (struct *uwbmac_msg**) – Message being written.
- **tag** (int) – Tag in the payload.
- **value** (int16_t) – Value to add.

2.1.152.1 Return

QERR_SUCCESS or error.

2.1.153 uwbmac_writer_add_s32

enum qerr **uwbmac_writer_add_s32**(struct *uwbmac_msg* *msg, int tag, int32_t value)

Add a signed 32-bit integer to the message.

Parameters

- **msg** (struct *uwbmac_msg**) – Message being written.
- **tag** (int) – Tag in the payload.
- **value** (int32_t) – Value to add.

2.1.153.1 Return

QERR_SUCCESS or error.

2.1.154 uwbmac_writer_add_s64

enum qerr **uwbmac_writer_add_s64**(struct *uwbmac_msg* *msg, int tag, int64_t value)

Add a signed 64-bit integer to the message.

Parameters

- **msg** (struct *uwbmac_msg**) – Message being written.
- **tag** (int) – Tag in the payload.
- **value** (int64_t) – Value to add.

2.1.154.1 Return

QERR_SUCCESS or error.

2.1.155 uwbmac_writer_add_u8

enum qerr **uwbmac_writer_add_u8**(struct *uwbmac_msg* *msg, int tag, uint8_t value)

Add a unsigned 8-bit integer to the message.

Parameters

- **msg** (struct *uwbmac_msg**) – Message being written.
- **tag** (int) – Tag in the payload.
- **value** (uint8_t) – Value to add.

2.1.155.1 Return

QERR_SUCCESS or error.

2.1.156 uwbmac_writer_add_u16

enum qerr **uwbmac_writer_add_u16**(struct *uwbmac_msg* *msg, int tag, uint16_t value)

Add a unsigned 16-bit integer to the message.

Parameters

- **msg** (struct *uwbmac_msg**) – Message being written.
- **tag** (int) – Tag in the payload.
- **value** (uint16_t) – Value to add.

2.1.156.1 Return

QERR_SUCCESS or error.

2.1.157 uwbmac_writer_add_u32

enum qerr **uwbmac_writer_add_u32**(struct *uwbmac_msg* *msg, int tag, uint32_t value)

Add a unsigned 32-bit integer to the message.

Parameters

- **msg** (struct *uwbmac_msg**) – Message being written.
- **tag** (int) – Tag in the payload.
- **value** (uint32_t) – Value to add.

2.1.157.1 Return

QERR_SUCCESS or error.

2.1.158 uwbmac_writer_add_u64

enum qerr **uwbmac_writer_add_u64**(struct *uwbmac_msg* *msg, int tag, uint64_t value)

Add a unsigned 64-bit integer to the message.

Parameters

- **msg** (struct *uwbmac_msg**) – Message being written.
- **tag** (int) – Tag in the payload.
- **value** (uint64_t) – Value to add.

2.1.158.1 Return

QERR_SUCCESS or error.

2.1.159 uwbmac_writer_add_string

enum qerr **`uwbmac_writer_add_string`**(struct *uwbmac_msg* *msg, int tag, const char *value)

Add a string to the message.

Parameters

- **`msg`** (`struct uwbmac_msg*`) – Message being written.
- **`tag`** (`int`) – Tag in the payload.
- **`value`** (`const char*`) – Pointer to the string to add.

2.1.159.1 Return

QERR_SUCCESS or error.

2.1.160 uwbmac_writer_add_binary

enum qerr **`uwbmac_writer_add_binary`**(struct *uwbmac_msg* *msg, int tag, const void *data, int length)

Add a binary object to the message.

Parameters

- **`msg`** (`struct uwbmac_msg*`) – Message being written.
- **`tag`** (`int`) – Tag in the payload.
- **`data`** (`const void*`) – Pointer to the object to add.
- **`length`** (`int`) – Length of the object to add.

2.1.160.1 Return

QERR_SUCCESS or error.

2.1.161 uwbmac_writer_start_nested

enum qerr **`uwbmac_writer_start_nested`**(struct *uwbmac_msg* *msg, int tag, struct *uwbmac_msg* *nested)

Start adding a nested payload to the message.

Parameters

- **`msg`** (`struct uwbmac_msg*`) – Message being written.
- **`tag`** (`int`) – Tag in the payload.
- **`nested`** (`struct uwbmac_msg*`) – Pointer to the structure to keep internal data.

2.1.161.1 Return

QERR_SUCCESS or error.

2.1.162 uwbmac_writer_end_nested

enum qerr **uwbmac_writer_end_nested**(struct *uwbmac_msg* *msg, struct *uwbmac_msg* *nested)

Stop adding a nested payload to the message.

Parameters

- **msg** (struct *uwbmac_msg**) – Message being written.
- **nested** (struct *uwbmac_msg**) – Pointer to the structure to keep internal data.

2.1.162.1 Return

QERR_SUCCESS or error.

2.1.163 uwbmac_writer_add_singleton_map

enum qerr **uwbmac_writer_add_singleton_map**(struct *uwbmac_msg* *msg)

Add a map containing a single pair to the message.

Parameters

- **msg** (struct *uwbmac_msg**) – Message being written.

2.1.163.1 Description

This function is only meant to be used at the beginning of an empty message, to produce a well-formed CBOR payload.

2.1.163.2 Return

QERR_SUCCESS or error. QERR_ENOTSUP if the message is not empty.

2.2 UWBMAC embedded API

2.2.1 uwbmac_device_state_report

void **uwbmac_device_state_report**(struct ieee802154_hw *hw, enum *uwbmac_device_state* state)

Report a device state change.

Parameters

- **hw** (struct ieee802154_hw*) – Pointer to MCPS hw instance.
- **state** (enum *uwbmac_device_state*) – New device state.

2.2.2 uwbmac_region_call_reply

enum qerr **uwbmac_region_call_reply**(struct ieee802154_hw *hw, struct sk_buff *reply)

Reply to a region call.

Parameters

- **hw** (`struct ieee802154_hw*`) – Pointer to MCPS hw instance.
- **reply** (`struct sk_buff*`) – Reply message.

2.2.2.1 Return

QERR_SUCCESS or error.

2.2.3 uwbmac_event_report

enum qerr **uwbmac_event_report**(struct ieee802154_hw *hw, uint32_t port_id, struct sk_buff *report)

Report an event.

Parameters

- **hw** (`struct ieee802154_hw*`) – Pointer to MCPS hw instance.
- **port_id** (`uint32_t`) – Port id to use to notify upper layer.
- **report** (`struct sk_buff*`) – Event report.

2.2.3.1 Return

QERR_SUCCESS or error.

2.2.4 uwbmac_testmode_reply

enum qerr **uwbmac_testmode_reply**(struct ieee802154_hw *hw, struct uwbmac_buf *reply)

Reply to a testmode call.

Parameters

- **hw** (`struct ieee802154_hw*`) – Pointer to MCPS hw instance.
- **reply** (`struct uwbmac_buf*`) – Reply message.

2.2.4.1 NOTE

This method is only used by embedded flavor.

2.2.4.2 Return

QERR_SUCCESS or error.

2.3 FiRa helper API

2.3.1 struct measurement_sequence

struct **measurement_sequence**

Fira measurement sequence.

2.3.1.1 Definition

```
struct measurement_sequence {
    size_t n_steps;
    struct fira_measurement_sequence_step steps[FIRA_MEASUREMENT_SEQUENCE_STEP_MAX];
}
```

2.3.1.2 Members

n_steps
Number of steps in the schedule.

steps
Steps of the schedule.

2.3.1.3 Description

This structure contains the measurement sequence executed by the region.

2.3.2 struct session_parameters

struct **session_parameters**

Fira session parameters.

2.3.2.1 Definition

```
struct session_parameters {
    uint8_t device_type;
    uint8_t device_role;
    uint8_t ranging_round_usage;
    uint8_t sts_config;
    uint8_t multi_node_mode;
    uint16_t short_addr;
    uint16_t destination_short_address[FIRA_RESPONDERS_MAX];
    int n_destination_short_address;
    uint64_t time0_ns;
    uint32_t slot_duration_rstu;
```

(continues on next page)

(continued from previous page)

```
    uint32_t round_duration_slots;
    uint32_t block_duration_ms;
    uint32_t block_stride_length;
    bool round_hopping;
    uint8_t priority;
    uint8_t mac_address_mode;
    uint8_t ranging_round_control;
    uint8_t schedule_mode;
    uint16_t max_number_of_measurements;
    uint32_t max_rr_retry;
    uint8_t channel_number;
    uint8_t preamble_code_index;
    uint8_t rframe_config;
    uint8_t preamble_duration;
    uint8_t sfd_id;
    uint8_t psdu_data_rate;
    uint8_t phr_data_rate;
    union {
        struct {
            uint8_t static_sts_iv[FIRA_STATIC_STS_IV_SIZE];
            uint8_t vendor_id[FIRA_VENDOR_ID_SIZE];
        } ;
        uint8_t vupper64[FIRA_VUPPER64_SIZE];
    } ;
    uint8_t key_rotation;
    uint8_t key_rotation_rate;
    uint32_t sub_session_id;
    uint8_t report_rssi;
    uint8_t result_report_config;
    uint8_t link_layer_mode;
    uint8_t mac_fcs_type;
    uint8_t prf_mode;
    uint8_t cap_size_min;
    uint8_t cap_size_max;
    uint8_t number_of_sts_segments;
    struct measurement_sequence meas_seq;
    bool enable_diagnostics;
    uint32_t diags_frame_reports_fields;
    uint8_t sts_length;
    uint8_t min_frames_per_rr;
    uint16_t mtu_size;
    uint8_t inter_frame_interval_ms;
    uint8_t owr_aoa_measurement_ntf_period;
    uint8_t session_info_ntf_config;
    uint32_t near_proximity_config_cm;
    uint32_t far_proximity_config_cm;
    int32_t lower_aoa_bound_config_azimuth_2pi;
    int32_t upper_aoa_bound_config_azimuth_2pi;
    int16_t lower_aoa_bound_config_elevation_2pi;
    int16_t upper_aoa_bound_config_elevation_2pi;
    uint8_t termination_count;
}
```

2.3.2.2 Members

device_type

Type of the device.

Possible values:

- 0x00: Controlee.
- 0x01: Controller.

See `enum quwbs_fbs_device_type`.

device_role

Role played by the device.

Current implementation does not support decorrelation between the device's role and the device's type. The controller can only behave as the initiator and the controlee can only behave as responder.

Possible values:

- 0x00: Responder.
- 0x01: Initiator.
- 0x02: UT-Synchronization Anchor. *[Not supported in QM33 SDK]*
- 0x03: UT-Anchor. *[Not supported in QM33 SDK]*
- 0x04: UT-Tag. *[Not supported in QM33 SDK]*
- 0x05: Advertiser. *[Not supported in QM33 SDK]*
- 0x06: Observer. *[Not supported in QM33 SDK]*
- 0x07: DT-Anchor. *[Not supported in QM33 SDK]*
- 0x08: DT-Tag. *[Not supported in QM33 SDK]*

See `enum quwbs_fbs_device_role`.

ranging_round_usage

The ranging mode used during a round.

Possible values:

- 0x00: OWR UL-TDoA. *[Not supported in QM33 SDK]*
- 0x01: SS-TWR with Deferred Mode.
- 0x02: DS-TWR with Deferred Mode.
- 0x03: SS-TWR with Non-deferred Mode.
- 0x04: DS-TWR with Non-deferred Mode.
- 0x05: OWR DL-TDoA. *[Not supported in QM33 SDK]*
- 0x06: OWR for AoA. *[Not supported in QM33 SDK]*
- 0x07: eSS-TWR with Non-deferred Mode for Contention-based ranging. *[Not supported in QM33 SDK]*
- 0x08: aDS-TWR with Non-deferred Mode for Contention-based ranging. *[Not supported in QM33 SDK]*

See *`enum fira_ranging_round_usage`*.

sts_config
It configures how system shall generate the STS.

Possible values:

- 0x00: Static STS (default).
- 0x01: Dynamic STS. *[Not supported in QM33 SDK]*
- 0x02: Dynamic STS - Responder Specific Sub-session Key. *[Not supported in QM33 SDK]*
- 0x03: Provisioned STS.
- 0x04: Provisioned STS - Responder Specific Sub-session Key.

See `enum fbs_sts_mode`.

multi_node_mode

The multi-node mode used during a round.

Possible values:

- 0x00: One-to-One.
- 0x01: One-to-Many.

See enum *`struct fira_multi_node_mode`*.

short_addr
Short address of the local device.

destination_short_address
Array of destination short addresses.

n_destination_short_address
Number of destination short addresses.

time0_ns
Absolute value of the initiation time in nanoseconds.

slot_duration_rstu
Duration of a slot in RSTU (1200RSTU=1ms).

round_duration_slots
Number of slots per ranging round.

block_duration_ms
Block size in unit of 1200 RSTU (same as ms).

block_stride_length
Number of blocks to stride.

round_hopping
Enable FiRa round hopping.

priority
Priority of the session.

mac_address_mode
MAC addressing mode.

ranging_round_control
Bit map of the following.

- b0: ranging result report phase is disabled(0) or enabled(1).
- b1: Control Message is sent in band(1) or not (0, not supported).

- b2: Control Message is sent separately(0) or piggybacked to RIM(1).

schedule_mode
Scheduling mode for the ranging session.

Possible values:

- 0x00 - Contention-based ranging. *[Not supported in QM33 SDK]*
- 0x01 - Time-scheduled ranging.
- 0x02 - Hybrid-based ranging. *[Not supported in QM33 SDK]*

max_number_of_measurements
Max number of measurements

max_rr_retry

Number of failed ranging round attempts before stopping the session.

The value zero disable the feature.

channel_number
UWB channel for this session.

preamble_code_index

UWB preamble code index.

Possible values:

- 9-24: BPRF
- 25-32: HPRF *[Not supported in QM33 SDK]*

rframe_config

The configuration of the frame.

see enum *`struct fira_rframe_config`*.

preamble_duration

Possible values:

- 0x00: 32 symbols *[Not supported in QM33 SDK]*
- 0x01: 64 symbols (default)

See *`enum fira_preamble_duration`*.

sfd_id

Possible values:

- 0 or 2 in BPRF
- 1-4 in HPRF *[Not supported in QM33 SDK]*

See *`enum fira_sfd_id`*.

psdu_data_rate

Possible values:

- 0: 6.81Mbps (default)
- 1: 7.80 Mbps *[Not supported in QM33 SDK]*
- 2: 27.2 Mbps *[Not supported in QM33 SDK]*
- 3: 31.2 Mbps *[Not supported in QM33 SDK]*

See `enum fira_psdu_data_rate`.

phr_data_rate

Possible values:

- 0: 850 kbit/s.
- 1: 6.81 Mbit/s.

See `enum fira_phr_data_rate`.

{unnamed_union}
anonymous

{unnamed_struct}
anonymous

static_sts_iv
Static STS IV used in vUpper64.

vendor_id
Vendor ID used in vUpper64.

vupper64
vUpper64 used during Static STS ranging.

key_rotation
Enable/disable key rotation feature during Dynamic *[Not supported in QM33 SDK]* or Provisioned STS ranging.

Possible values:

- false: No key rotation.
- true: Key rotation enabled and period set by key_rotation_rate.

key_rotation_rate
Defines n, with 2^n being the rotation rate of some keys used during Dynamic *[Not supported in QM33 SDK]* or Provisioned STS Ranging, n shall be in the range of 0<=n<=15.

sub_session_id
Sub-session id for the controlee device. This configuration is applicable if STS_CONFIG is set to 0x02 or 0x04.

report_rssi
Activate rssi report

Possible values:

- 0: no rssi report
- 1: activate rssi report

result_report_config
Configure report information.

- b0: report ToF in result message, disabled(0) or enabled(1, default)
- b1: report AoA azimuth in result message, disabled (0, default) or enabled (1)
- b2: report AoA elevation in result message, disabled (0, default) or enabled (1)
- b3: report AoA FOM in result message, disabled (0, default) or enabled (1)

link_layer_mode
Used to define link layer behavior.

Possible values:

- 0x00: Bypass mode (default).

- 0x01: Connection less. *[Not supported in QM33 SDK]*
- Values 0x02 to 0xFF: RFU.

mac_fcs_type

[NOT IMPLEMENTED] The length of the Frame Check Sequence in the session.

Possible values:

- 0x00: CRC 16 (default)
- 0x01: CRC 32
- Values 0x02 to 0xFF: RFU

This parameter is not used in the current implementation.

See *enum fira_mac_fcs_type*.

prf_mode

Possible values:

- 0x00: 62.4 MHz PRF. BPRF mode (default)
- 0x01: 124.8 MHz PRF. HPRF mode. *[Not supported in QM33 SDK]*
- 0x02: 249.6 MHz PRF. HPRF mode with data rate 27.2 and 31.2 Mbps. *[Not supported in QM33 SDK]*

See *enum fira_prf_mode*.

cap_size_min

[Not supported in QM33 SDK] Contention access period minimum value.

Default: 5

cap_size_max

[Not supported in QM33 SDK] Contention access period maximum value.

Default: round_duration_slots - 1

number_of_sts_segments

[NOT IMPLEMENTED] Number of STS segments.

Possible values:

- 0x01: 1 STS Segment (default)
- 0x02: 2 STS Segments (HPRF only) *[Not supported in QM33 SDK]*
- 0x03: 3 STS Segments (HPRF only) *[Not supported in QM33 SDK]*
- 0x04: 4 STS Segments (HPRF only) *[Not supported in QM33 SDK]*
- Values 0x05 to 0xFF: RFU

This parameter is not used in the current implementation.

meas_seq

[Not supported in QM33 SDK] Sequence of measurement sequence steps, configures the Antenna Flexibility features.

enable_diagnostics

Activate the diagnostics for each round.

diags_frame_reports_fields

Select the fields to activate in the frame reports stored in the diagnostics. Applicable only when `enable_diagnostics` is set to true.

sts_length

Number of symbols in a STS segment.

Possible values:

- 0x00: 32 symbols
- 0x01: 64 symbols (default)
- 0x02: 128 symbols
- Values 0x03 to 0xFF: RFU

min_frames_per_rr

[Not supported in QM33 SDK] Minimal number of frames to be transmitted in OWR for AoA ranging round (block).

This parameter is only used in OWR for AoA Mode, see ranging_round_usage paramater

mtu_size

[Not supported in QM33 SDK] Maximum Transfer Unit, max size allowed to be transmitted in frame. The value shall be restricted to the maximum possible MTU size of the given frame which includes MHR, Variable IE size and FCS size.

inter_frame_interval_ms

[Not supported in QM33 SDK] Interval between RFRAMES transmitted in OWR for AoA (in units of 1200 RSTU)

This parameter is only used in OWR for AoA Mode, see ranging_round_usage paramater

owr_aoa_measurement_ntf_period

[Not supported in QM33 SDK] Configure period of OWR for AoA measurement notifications.

Possible values:

- 0x00 = SESSION_INFO_NTF sent for every received OWR Advertisement frame (default)
- 0x01 = SESSION_INFO_NTF sent once after MIN_FRAMES_PER_RR number of AoA measurements are aggregated

session_info_ntf_config

[Not supported in QM33 SDK] Configure session info notification.

Possible values:

- 0x00 = Disable session info notification (ntf)
- 0x01 = Enable session info notification (default)
- 0x02 = Enable session info ntf while inside proximity range
- 0x03 = Enable session info ntf while inside AoA upper and lower bounds
- 0x04 = Enable session info ntf while inside AoA upper and lower bounds as well as inside proximity range
- 0x05 = Enable session info ntf only when entering or leaving proximity range
- 0x06 = Enable session info ntf only when entering or leaving AoA upper and lower bounds
- 0x07 = Enable session info ntf only when entering or leaving AoA upper and lower bounds as well as entering or leaving proximity range

near_proximity_config_cm

[Not supported in QM33 SDK] Lower bound in cm above which the ranging notifications should be enabled.

Applicable when session_info_ntf_config is set to 0x02, 0x04, 0x05 or 0x07. Should be less than or equal to far_proximity_config value.

far_proximity_config_cm

[Not supported in QM33 SDK] Upper bound in cm above which the ranging notifications should be disabled.

Applicable when session_info_ntf_config is set to 0x02, 0x04, 0x05 or 0x07. Should be greater than or equal to near_proximity_config value.

lower_aoa_bound_config_azimuth_2pi

[Not supported in QM33 SDK] Represent degrees.

Applicable when session_info_ntf_config is set to 0x03, 0x04, 0x06 or 0x07.

upper_aoa_bound_config_azimuth_2pi

[Not supported in QM33 SDK] Represent degrees.

Applicable when session_info_ntf_config is set to 0x03, 0x04, 0x06 or 0x07.

lower_aoa_bound_config_elevation_2pi

[Not supported in QM33 SDK] Represent degrees

Applicable when session_info_ntf_config is set to 0x03, 0x04, 0x06 or 0x07.

upper_aoa_bound_config_elevation_2pi

[Not supported in QM33 SDK] Represent degrees.

Applicable when session_info_ntf_config is set to 0x03, 0x04, 0x06 or 0x07.

termination_count

in band termination attempt count.

2.3.2.3 Description

This structure contains the session parameters sent to the Fira region. Current implementation does not use all the parameters defined below.

2.3.3 struct controlee_parameters

struct `controlee_parameters`

Controlee parameters.

2.3.3.1 Definition

```
struct controlee_parameters {
    uint32_t sub_session_id;
    uint16_t address;
    bool sub_session;
    uint8_t sub_session_key_len;
    uint8_t sub_session_key[FIRA_KEY_SIZE_MAX];
}
```

2.3.3.2 Members

sub_session_id
Sub-session id for the controlee device.

address
Controlee short address.

sub_session
To indicate whether or not the controlee has a sub-session.

sub_session_key_len
Size of the sub-session key, either 16 or 32 bytes.

sub_session_key
Key used for sub-session's crypto calculations.

2.3.4 struct controlees_parameters

struct `controlees_parameters`
Controlees list parameters.

2.3.4.1 Definition

```
struct controlees_parameters {
    struct controlee_parameters controlees[FIRA_RESPONDERS_MAX];
    int n_controlees;
}
```

2.3.4.2 Members

controlees
List of controlees.

n_controlees
Number of controlees in the list.

2.3.5 struct dt_anchor_ranging_round_config

struct `dt_anchor_ranging_round_config`
[Not supported in QM33 SDK] Configuration parameters of the ranging round for DT-Anchor.

2.3.5.1 Definition

```
struct dt_anchor_ranging_round_config {
    uint8_t round_index;
    uint8_t acting_role;
    uint8_t n_responders;
    bool are_slots_present;
    uint16_t responders[FIRA_RESPONDERS_MAX];
    uint8_t slots[FIRA_RESPONDERS_MAX];
}
```

2.3.5.2 Members

round_index
Round index.

acting_role
Acting role (Initiator or Responder).

n_responders
Total number of DT-Anchor Responders for this ranging round (applicable when the acting role is Initiator).

are_slots_present
Flag indicating if explicit slot scheduling will follow (applicable when the acting role is Initiator).

responders
Short addresses of DT-Anchor Responders for this ranging round (applicable when the acting role is Initiator).

slots
Slots for Response DTMs for consecutive DT-Anchor Responders (explicit slot scheduling, applicable when the acting role is Initiator).

2.3.6 struct update_dt_anchor_ranging_rounds_cmd

struct **update_dt_anchor_ranging_rounds_cmd**

[Not supported in QM33 SDK] Request to MAC with configuration of the ranging rounds for DT-Anchor.

2.3.6.1 Definition

```
struct update_dt_anchor_ranging_rounds_cmd {
    int n_ranging_rounds;
    struct dt_anchor_ranging_round_config *ranging_rounds;
}
```

2.3.6.2 Members

n_ranging_rounds
Total number of round configurations.

ranging_rounds
Configuration parameters per ranging round.

2.3.7 struct update_dt_anchor_ranging_rounds_rsp

struct **update_dt_anchor_ranging_rounds_rsp**

[Not supported in QM33 SDK] Response from MAC including indexes of ranging rounds which failed to be configured for DT-Anchor.

2.3.7.1 Definition

```
struct update_dt_anchor_ranging_rounds_rsp {
    enum quwbs_fbs_status status;
    int n_round_indexes;
    uint8_t round_indexes[FIRA_DT_ANCHOR_MAX_ACTIVE_RR];
}
```

2.3.7.2 Members

status
Status of the config_rsp.

n_round_indexes
Number of failing rounds.

round_indexes
Failing round indexes.

2.3.8 struct dt_tag_ranging_rounds_config

struct **dt_tag_ranging_rounds_config**
[Not supported in QM33 SDK] Configuration parameters of the ranging rounds for DT-Tag.

2.3.8.1 Definition

```
struct dt_tag_ranging_rounds_config {
    uint8_t *round_indexes;
    int n_round_indexes;
}
```

2.3.8.2 Members

round_indexes
Round indexes.

n_round_indexes
Total number of round indexes.

2.3.9 struct dt_tag_round_indexes_rsp

struct **dt_tag_round_indexes_rsp**
[Not supported in QM33 SDK] Indexes of ranging rounds which failed to be configured for DT-Tag.

2.3.9.1 Definition

```
struct dt_tag_round_indexes_rsp {
    enum quwbs_fbs_status status;
    int n_round_indexes;
    uint8_t round_indexes[FBS_DT_TAG_MAX_ACTIVE_RR];
}
```

2.3.9.2 Members

status
    Status of the config_rsp.

n_round_indexes
    Total number of round indexes.

round_indexes
    Round indexes.

2.3.10 enum aoa_measurements_index

enum **aoa_measurements_index**
    AOA measurements.

2.3.10.1 Definition

```
enum aoa_measurements_index {
    FIRA_HELPER_AOA_AZIMUTH,
    FIRA_HELPER_AOA,
    FIRA_HELPER_AOA_ELEVATION,
    FIRA_HELPER_AOA_NB
};
```

2.3.10.2 Constants

FIRA_HELPER_AOA_AZIMUTH
    Retrieve AOA azimuth.

FIRA_HELPER_AOA
    Retrieve AOA (same as azimuth).

FIRA_HELPER_AOA_ELEVATION
    Retrieve AOA elevation. *[Not supported in QM33 SDK]*

FIRA_HELPER_AOA_NB
    Enum members number.

2.3.11 struct aoa_measurements

struct **aoa_measurements**

Fira Angle of Arrival measurements.

2.3.11.1 Definition

```
struct aoa_measurements {
    uint8_t rx_antenna_pair;
    uint8_t aoa_fom_100;
    int16_t aoa_2pi;
    int16_t pdoa_2pi;
}
```

2.3.11.2 Members

rx_antenna_pair
Antenna pair index.

aoa_fom_100
Estimation of local AoA reliability.

aoa_2pi
Estimation of reception angle.

pdoa_2pi
Estimation of reception phase difference.

2.3.11.3 Description

Contains the different results of the AOA measurements.

2.3.12 struct fira_twr_measurements

struct **fira_twr_measurements**

Fira ranging measurements.

2.3.12.1 Definition

```
struct fira_twr_measurements {
    uint16_t short_addr;
    uint8_t status;
    uint8_t slot_index;
    bool stopped;
    uint8_t nlos;
    int32_t distance_cm;
    int16_t remote_aoa_azimuth_2pi;
    int16_t remote_aoa_elevation_pi;
    uint8_t remote_aoa_azimuth_fom_100;
    uint8_t remote_aoa_elevation_fom_100;
```

(continues on next page)

(continued from previous page)

```
    struct aoa_measurements local_aoa_measurements[FIRA_HELPER_AOA_NB];
    uint8_t rssi;
}
```

2.3.12.2 Members

short_addr
Address of the participating device.

status
Zero if ok, or error reason.

slot_index
In case of error, slot index where the error was detected.

stopped
Ranging was stopped as requested [controller only].

nlos
Indicates if the ranging measurement was in Line of Sight (LoS) or Non-Line of Sight (NLoS): 0x00 = LoS, 0x01 = NLoS, 0xFF = Unable to determine.

distance_cm
Distance in cm.

remote_aoa_azimuth_2pi
Estimation of reception angle in the azimuth of the participating device.

remote_aoa_elevation_pi
Estimation of reception angle in the elevation of the participating device.

remote_aoa_azimuth_fom_100
Estimation of azimuth reliability of the participating device.

remote_aoa_elevation_fom_100
Estimation of elevation of the participating device.

local_aoa_measurements
Table of estimations of local measurements.

rssi
Computed rssi

2.3.13 struct fira_ranging_info

struct **fira_ranging_info**
Common information on the ranging result.

2.3.13.1 Definition

```
struct fira_ranging_info {
    uint32_t session_handle;
    uint32_t sequence_number;
    uint32_t block_index;
    uint32_t ranging_interval_ms;
    uint64_t timestamp_ns;
    struct diagnostic_info *diagnostic;
    struct uwbmac_buf *psdus_report;
}
```

2.3.13.2 Members

session_handle
Session handle of the ranging result.

sequence_number
Session notification counter.

block_index
Current block index.

ranging_interval_ms
Current ranging interval in unit of ms. formula: (block size * (stride + 1))

timestamp_ns

[NOT IMPLEMENTED] Timestamp in nanoseconds in the CLOCK_MONOTONIC time reference.

The current implementation does not provide any timestamp.

diagnostic
Debug informations

psdus_report
Report containing all the psdus.

2.3.14 struct fira_twr_ranging_results

struct `fira_twr_ranging_results`
Ranging results for Fira SS-TWR/DS-TWR.

2.3.14.1 Definition

```
struct fira_twr_ranging_results {
    struct fira_ranging_info *info;
    int n_measurements;
    struct fira_twr_measurements measurements[FIRA_RESPONDERS_MAX];
}
```

2.3.14.2 Members

info
: Common information on this ranging.

n_measurements

: Number of measurements stored in the measurements table.

measurements
: Ranging measurements information.

2.3.15 struct fira_owr_aoa_measurements

struct **fira_owr_aoa_measurements**
: *[Not supported in QM33 SDK]* Ranging measurement for Fira OWR AoA.

2.3.15.1 Definition

```
struct fira_owr_aoa_measurements {
    uint16_t short_addr;
    uint8_t status;
    uint8_t nlos;
    uint8_t frame_sequence_number;
    uint16_t block_index;
    struct aoa_measurements local_aoa_measurements[FIRA_HELPER_AOA_NB];
}
```

2.3.15.2 Members

short_addr
: Address of the participating device.

status
: Zero if ok, or error reason.

nlos
: Indicates if the reception of the message was in Line of Sight (LoS) or Non-Line of Sight (NLoS): 0x00 = LoS, 0x01 = NLoS, 0xFF = Unable to determine.

frame_sequence_number
: Sequence number as received in MHR.

block_index

: Block Index number as received in the OWR message from the Advertiser.

local_aoa_measurements
: Table of estimations of local measurements.

2.3.16 struct fira_owr_aoa_ranging_results

struct **fira_owr_aoa_ranging_results**

[Not supported in QM33 SDK] Ranging results for Fira OWR AOA.

2.3.16.1 Definition

```
struct fira_owr_aoa_ranging_results {
    struct fira_ranging_info *info;
    int n_measurements;
    struct fira_owr_aoa_measurements measurements[FIRA_OWR_AOA_MEASUREMENTS_MAX];
}
```

2.3.16.2 Members

info

Common information on this ranging.

n_measurements

Number of measurements stored in the measurements table.

measurements

Ranging measurements information.

2.3.17 struct fira_ul_tdoa_ranging_results

struct **fira_ul_tdoa_ranging_results**

[Not supported in QM33 SDK] Ranging results for FiRa UL-TDoA. Will be extended with implementation of UT-Anchor.

2.3.17.1 Definition

```
struct fira_ul_tdoa_ranging_results {
    struct fira_ranging_info *info;
}
```

2.3.17.2 Members

info

Common information on this ranging.

2.3.18 struct fira_dl_tdoa_measurements

struct **fira_dl_tdoa_measurements**

[Not supported in QM33 SDK] DL-TDOA ranging measurements.

2.3.18.1 Definition

```
struct fira_dl_tdoa_measurements {
    struct fira_dl_tdoa_measurements *next;
    uint16_t short_addr;
    enum quwbs_fbs_status status;
    enum fira_owr_message_type message_type;
    enum fira_owr_dtm_timestamp_type tx_timestamp_type;
    enum fira_owr_dtm_timestamp_len tx_timestamp_len;
    enum fira_owr_dtm_timestamp_len rx_timestamp_len;
    enum fira_dt_location_coord_system_type anchor_location_type;
    bool anchor_location_present;
    uint8_t active_ranging_round_indexes_len;
    uint8_t round_index;
    uint16_t block_index;
    int16_t local_aoa_azimuth_2pi;
    int16_t local_aoa_elevation_2pi;
    uint8_t local_aoa_azimuth_fom;
    uint8_t local_aoa_elevation_fom;
    uint8_t rx_rssi;
    uint8_t nlos;
    uint16_t local_cfo;
    uint16_t remote_cfo;
    uint64_t tx_timestamp_rctu;
    uint64_t rx_timestamp_rctu;
    uint32_t initiator_reply_time_rctu;
    uint32_t responder_reply_time_rctu;
    uint8_t anchor_location[FIRA_DL_TDOA_ANCHOR_LOCATION_SIZE_MAX];
    uint8_t active_ranging_round_indexes[FIRA_DL_TDOA_MAX_ROUNDS_PER_BLOCK];
    uint16_t initiator_responder_tof_rctu;
}
```

2.3.18.2 Members

next
Pointer on next measurements if there is one, or NULL.

short_addr
Address of the participating device.

status
Zero if ok, or error reason. See enum quwbs_fbs_status for all error codes.

message_type
Type of the message which has been received.

tx_timestamp_type
Type of the TX timestamp (local time base vs common time base) included in the received message.

tx_timestamp_len
Length of the TX timestamp (40-bit vs 64-bit) included in the received message.

rx_timestamp_len
Length of the TX timestamp (40-bit vs 64-bit) calculated during the reception of the received message.

anchor_location_type
Type of the coordinate system of DT-Anchor location (0: WGS84, 1: relative) (if included).

anchor_location_present
True when the information about DT-Anchor location is included in the measurement, false otherwise.

active_ranging_round_indexes_len
Number of active ranging round indexes included in the measurement.

round_index
Index of the current ranging round.

block_index
Index of the current ranging block.

local_aoa_azimuth_2pi
AoA Azimuth in degrees measured by the DT-Tag during the reception (encoded as Q9.7).

local_aoa_elevation_2pi
AoA Elevation in degrees measured by the DT-Tag during the reception (encoded as Q9.7).

local_aoa_azimuth_fom
Reliability of the estimated AoA Azimuth measured by the DT-Tag during the reception (range: 0-100).

local_aoa_elevation_fom
Reliability of the estimated AoA Elevation measured by the DT-Tag during the reception (range: 0-100).

rx_rssi
RSSI measured by the DT-Tag during the reception (encoded as Q7.1).

nlos
Indicates if the reception of the message was in Line of Sight (LoS) or Non-Line of Sight (NLoS): 0x00 = LoS, 0x01 = NLoS, 0xFF = Unable to determine.

local_cfo
Clock frequency offset measured locally with respect to the DT-Anchor that sent the message received (encoded as Q6.10).

remote_cfo
Clock frequency offset of a Responder DT-Anchor with respect to the Initiator DT-Anchor of the ranging round as included in the received message (encoded as Q6.10).

tx_timestamp_rctu
TX timestamp included in the received message (unit: RCTU).

rx_timestamp_rctu
RX timestamp calculated during the reception of the received message (unit: RCTU).

initiator_reply_time_rctu
Reply time of the Initiator DT-Anchor measured between the reception of Response DTM and the transmission of Final DTM (used only in DS-TWR, unit: RCTU).

responder_reply_time_rctu
Reply time of the Responder DT-Anchor measured between the reception of Poll DTM and the transmission of Response DTM (unit: RCTU).

anchor_location
Location coordinates of DT-Anchor that sent the message received.

active_ranging_round_indexes
List of active ranging round indexes in which the DT-Anchor that sent the message received participates.

initiator_responder_tof_rctu
Time of Flight measured between the Initiator DT-Anchor and the Responder DT-Anchor (for SS-TWR it's calculated by Initiator DT-Anchor and included in Poll DTM and for DS-TWR it's calculated by Responder DT-Anchor and included in Response DTM, unit: RCTU)

2.3.19 struct fira_dl_tdoa_ranging_results

struct `fira_dl_tdoa_ranging_results`
[Not supported in QM33 SDK] Ranging results for Fira DL-TDOA.

2.3.19.1 Definition

```
struct fira_dl_tdoa_ranging_results {
    struct fira_ranging_info *info;
    int n_measurements;
    struct fira_dl_tdoa_measurements *measurements;
}
```

2.3.19.2 Members

info
Common information on this ranging.

n_measurements
Number of measurements stored in the measurements table.

measurements
Linked list of the DL-TDOA measurements or NULL.

2.3.20 struct controlee_status

struct `controlee_status`
Controlee addition/deletion notification status.

2.3.20.1 Definition

```
struct controlee_status {
    uint16_t short_address;
    uint32_t sub_session_id;
    uint8_t status_code;
}
```

2.3.20.2 Members

short_address
Controlee short address.

sub_session_id
Sub-session id of the current controlee.

status_code
See *enum fira_multicast_update_status*.

2.3.21 struct fira_session_multicast_list_ntf_content

struct **fira_session_multicast_list_ntf_content**
Necessary content to fill a session update controller multicast list notification.

2.3.21.1 Definition

```
struct fira_session_multicast_list_ntf_content {
    uint32_t session_handle;
    uint8_t remaining_multicast_list_size;
    struct controlee_status controlees[FIRA_RESPONDERS_MAX];
    uint8_t n_controlees;
}
```

2.3.21.2 Members

session_handle
Session handle.

remaining_multicast_list_size
New available size in the multicast list. Maximum size is defined by FIRA_RESPONDERS_MAX.

controlees
List of controlees with their corresponding multicast list update status.

n_controlees
Number of controlees in the previous list.

2.3.22 struct data_credit_ntf_content

struct **data_credit_ntf_content**
[Not supported in QM33 SDK] Fira DATA_CREDIT_NFT content.

2.3.22.1 Definition

```
struct data_credit_ntf_content {
    uint32_t session_handle;
    uint8_t credit_avail;
}
```

2.3.22.2 Members

session_handle
Session handle.

credit_avail
Credit availability 0x00 Credit is not available 0x01 Credit is available

2.3.23 struct data_transfer_status_ntf_content

struct **data_transfer_status_ntf_content**
[Not supported in QM33 SDK] Fira SESSION_DATA_TRANSFER_STATUS_NTF content.

2.3.23.1 Definition

```
struct data_transfer_status_ntf_content {
    uint32_t session_handle;
    uint16_t uci_seq_nr;
    uint8_t status;
    uint8_t tx_count;
}
```

2.3.23.2 Members

session_handle
Session handle.

uci_seq_nr
The Sequence Number identifying the UCI Data Message this NTF is for.

status
Status Code. See `enum uci_data_transfer_status_code`.

tx_count
Indicates the number of times Application Data with the same UCI Sequence Number has been transmitted.

2.3.24 struct data_message_content

struct **data_message_content**

[Not supported in QM33 SDK] Fira DATA_MESSAGE_SND and DATA_MESSAGE_RCV content.

2.3.24.1 Definition

```
struct data_message_content {
    uint32_t session_handle;
    uint16_t short_addr;
    uint16_t uci_seq_nr;
    uint8_t status;
    uint8_t data_segment_info;
    uint16_t data_len;
    uint8_t *data;
}
```

2.3.24.2 Members

session_handle
Session handle.

short_addr
Short_addr. For DATA_MESSAGE_SND: short_addr of the Application Data recipient. For DATA_MESSAGE_RCV: short_addr of the sender of the Application Data.

uci_seq_nr
Sequence Number for the UCI Data Message.

status
Status. Applicable only in case of DATA_MESSAGE_RCV. 0x00 STATUS_SUCCESS 0x01 STATUS_ERROR 0x02 STATUS_UNKNOWN

data_segment_info
See *enum fira_data_segment_info*.

data_len
Length of the data.

data
Application Data.

2.3.25 struct fira_hus_controller_phase_config

struct **fira_hus_controller_phase_config**

[Not supported in QM33 SDK] Phase configuration parameters used by a HUS controller device to bind a secondary session to a primary session.

2.3.25.1 Definition

```
struct fira_hus_controller_phase_config {
    uint32_t session_id;
    uint16_t start_slot_index;
    uint16_t end_slot_index;
    uint16_t controller_short_addr;
    uint8_t control;
}
```

2.3.25.2 Members

session_id
Session id of the targeted phase.

start_slot_index
Slot index of the first slot of the phase.

end_slot_index
Slot index of the last slot of the phase.

controller_short_addr
MAC short address of the controller of the phase.

control
Information about the current phase. b0: 0 = Short addressing mode, 1 = Extended addressing mode. b1: 0 = CAP phase, 1 = CFP phase.

2.3.26 struct fira_hus_controller_config_cmd

struct **fira_hus_controller_config_cmd**

[Not supported in QM33 SDK] List of secondary sessions to bind to a primary session. Only applicable to a HUS controller device.

2.3.26.1 Definition

```
struct fira_hus_controller_config_cmd {
    uint64_t update_time_us;
    uint32_t session_handle;
    struct fira_hus_controller_phase_config *phase_list;
    uint8_t number_of_phases;
}
```

2.3.26.2 Members

update_time_us
Time in microseconds when this configuration shall be applied.

session_handle
Handle of the targeted session.

phase_list
List of CAP or CFP phases.

number_of_phases
Number of CAP or CFP phases in the HUS ranging round.

2.3.27 struct fira_hus_controlee_phase_config

struct **fira_hus_controlee_phase_config**
[Not supported in QM33 SDK] Phase configuration parameters used by a HUS controlee device to bind a secondary session to a primary session.

2.3.27.1 Definition

```
struct fira_hus_controlee_phase_config {
    uint32_t session_handle;
}
```

2.3.27.2 Members

session_handle
Session handle of the targeted phase.

2.3.28 struct fira_hus_controlee_config_cmd

struct **fira_hus_controlee_config_cmd**
[Not supported in QM33 SDK] Status of the configuration command binding secondary sessions to a primary session.

2.3.28.1 Definition

```
struct fira_hus_controlee_config_cmd {
    uint32_t session_handle;
    struct fira_hus_controlee_phase_config *phase_list;
    uint8_t number_of_phases;
}
```

2.3.28.2 Members

session_handle
Handle of the targeted session.

phase_list
List of CAP or CFP phases.

number_of_phases
Number of CAP or CFP phases in the HUS ranging round.

2.3.29 enum fira_helper_cb_type

enum **fira_helper_cb_type**
Callback type. See *struct fira_helper_notification_cb_t*

2.3.29.1 Definition

```
enum fira_helper_cb_type {
    FIRA_HELPER_CB_TYPE_UNSPEC,
    FIRA_HELPER_CB_TYPE_TWR_RANGE_NTF,
    FIRA_HELPER_CB_TYPE_OWR_AOA_NTF,
    FIRA_HELPER_CB_TYPE_UL_TDOA_NTF,
    FIRA_HELPER_CB_TYPE_DL_TDOA_NTF,
    FIRA_HELPER_CB_TYPE_SESSION_DATA_CREDIT_NTF,
    FIRA_HELPER_CB_TYPE_SESSION_DATA_TRANSFER_STATUS_NTF,
    FIRA_HELPER_CB_TYPE_DATA_MESSAGE_RCV,
    FIRA_HELPER_CB_TYPE_SESSION_STATUS_NTF,
    FIRA_HELPER_CB_TYPE_SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_NTF
};
```

2.3.29.2 Constants

FIRA_HELPER_CB_TYPE_UNSPEC
unspecified callback type

FIRA_HELPER_CB_TYPE_TWR_RANGE_NTF
Callback content is struct fira_twr_ranging_results*.

FIRA_HELPER_CB_TYPE_OWR_AOA_NTF
[Not supported in QM33 SDK] Callback content is struct fira_owr_aoa_ranging_results*.

FIRA_HELPER_CB_TYPE_UL_TDOA_NTF
[Not supported in QM33 SDK] Callback content is struct fira_ul_tdoa_ranging_results*.

FIRA_HELPER_CB_TYPE_DL_TDOA_NTF
[Not supported in QM33 SDK] Callback content is struct fira_dl_tdoa_ranging_results*.

FIRA_HELPER_CB_TYPE_SESSION_DATA_CREDIT_NTF
[Not supported in QM33 SDK] Callback content is struct data_credit_ntf_content*.

FIRA_HELPER_CB_TYPE_SESSION_DATA_TRANSFER_STATUS_NTF
[Not supported in QM33 SDK] Callback content is struct data_transfer_status_ntf_content*.

FIRA_HELPER_CB_TYPE_DATA_MESSAGE_RCV
[Not supported in QM33 SDK] Callback content is struct data_message_content*.

FIRA_HELPER_CB_TYPE_SESSION_STATUS_NTF
Callback content is struct fbs_helper_session_status_ntf*.

FIRA_HELPER_CB_TYPE_SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_NTF
Callback content is struct fira_session_multicast_list_ntf_content*.

2.3.30 typedef fira_helper_notification_cb_t

void **fira_helper_notification_cb_t**(enum *fira_helper_cb_type* cb_type, const void *content, void *user_data)

Notification callback type.

Parameters

- **cb_type** (enum *fira_helper_cb_type*) – Type of callback depending on exact message to be sent.
- **content** (const void*) – Generic content with results depending on the cb_type.
- **user_data** (void*) – User data pointer given to fira_helper_open.

2.3.30.1 Description

See enum fira_helper_cb_type documentation for more information on the content.

2.3.31 fira_helper_open

enum qerr **fira_helper_open**(struct fira_context *ctx, struct uwbmac_context *uwbmac, *fira_helper_notification_cb_t* notification_cb, const char *scheduler, int region_id, void *user_data)

Initialize the fira helper context.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **uwbmac** (struct uwbmac_context*) – UWB MAC context.
- **notification_cb** (*fira_helper_notification_cb_t*) – Callback function for notifications feedback.
- **scheduler** (const char*) – Scheduler name to use with the region.
- **region_id** (int) – Region identifier to associate with the region.
- **user_data** (void*) – User data pointer to give back in callback.

2.3.31.1 NOTE

This function must be called first. `fira_helper_close` must be called at the end of the application to ensure resources are freed. The channel will be managed by the helper, this means you should neither use uwbmac_channel_create nor uwbmac_channel_release.

2.3.31.2 Return

QERR_SUCCESS on success, an error otherwise.

2.3.32 fira_helper_close

void **fira_helper_close**(struct fira_context *ctx)

De-initialize the fira helper context.

Parameters

- **ctx** (`struct fira_context*`) – Fira helper context.

2.3.33 fira_helper_set_device_status_cb

enum qerr **fira_helper_set_device_status_cb**(struct fira_context *ctx, fbs_helper_device_status_ntf_cb cb)

Set the device status callback.

Parameters

- **ctx** (`struct fira_context*`) – Fira helper context.
- **cb** (`fbs_helper_device_status_ntf_cb`) – Callback for all device status notifications.

2.3.33.1 NOTE

Temporary api before we inverse dependancy with fbs_helper. Once this is done client will have to directly use fbs_helper_set_device_status_ntf_cb.

2.3.33.2 Return

QERR_SUCCESS on success, on error otherwise.

2.3.34 fira_helper_set_scheduler

enum qerr **fira_helper_set_scheduler**(struct fira_context *ctx)

Set the scheduler and open the MAC region.

Parameters

- **ctx** (`struct fira_context*`) – Fira helper context.

2.3.34.1 NOTE

This function must be called while the UWB MAC is stopped.

2.3.34.2 Return

QERR_SUCCESS on success, an error otherwise.

2.3.35 fira_helper_get_capabilities

enum qerr **fira_helper_get_capabilities**(struct fira_context *ctx, struct fira_capabilities *capabilites)

Get the FiRa region capabilities.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **capabilites** (struct fira_capabilities*) – Fira capabilites.

2.3.35.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.36 fira_helper_init_session

enum qerr **fira_helper_init_session**(struct fira_context *ctx, uint32_t session_id, enum quwbs_fbs_session_type session_type, struct fbs_session_init_rsp *rsp)

Initialize a fira session.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_id** (uint32_t) – Session identifier.
- **session_type** (enum quwbs_fbs_session_type) – Session type value.
- **rsp** (struct fbs_session_init_rsp*) – Session init response message information.

2.3.36.1 Description

This function must be called first to create and initialize the fira session.

2.3.36.2 Return

QERR_SUCCESS on success, an error otherwise.

2.3.37 fira_helper_start_session

enum qerr **fira_helper_start_session**(struct fira_context *ctx, uint32_t session_handle)

Start a fira session.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.

2.3.37.1 Description

This function must be called after fira session was initialized.

2.3.37.2 Return

QERR_SUCCESS on success, an error otherwise.

2.3.38 fira_helper_stop_session

enum qerr **fira_helper_stop_session**(struct fira_context *ctx, uint32_t session_handle)

Stop a fira session.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.

2.3.38.1 Description

This function stop the session ranging.

2.3.38.2 Return

QERR_SUCCESS on success, an error otherwise.

2.3.39 fira_helper_deinit_session

enum qerr **fira_helper_deinit_session**(struct fira_context *ctx, uint32_t session_handle)

Deinitialize a fira session.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.

2.3.39.1 Description

This function is called to free all memory allocated by the session.

2.3.39.2 Return

QERR_SUCCESS or QERR_EBUSY on success, an error otherwise.
The QERR_EBUSY is used to indicate that an active session has been deinit.

2.3.40 fira_helper_get_session_parameters

enum qerr **fira_helper_get_session_parameters**(struct fira_context *ctx, uint32_t session_handle, struct *session_parameters* *session_params)

Get session parameters.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **session_params** (struct *session_parameters**) – Session parameters.

2.3.40.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.41 fira_helper_session_get_count

enum qerr **fira_helper_session_get_count**(struct fira_context *ctx, int *count)

Get sessions count, the number of active and inactive sessions.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **count** (int*) – Session count.

2.3.41.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.42 fira_helper_session_get_state

enum qerr **fira_helper_session_get_state**(struct fira_context *ctx, uint32_t session_handle, int *state)

Get session state.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **state** (int*) – Session state.

2.3.42.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.43 fira_helper_get_ranging_count

enum qerr **fira_helper_get_ranging_count**(struct fira_context *ctx, uint32_t session_handle, struct fbs_ranging_count_rsp *rsp)

Get ranging count, the number of times ranging has been attempted during the session.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **rsp** (struct fbs_ranging_count_rsp*) – Ranging count response message information.

2.3.43.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.44 fira_helper_add_controlee

int **fira_helper_add_controlee**(struct fira_context *ctx, uint32_t session_handle, const struct *controlee_parameters* *controlee)

Add one controlee to a specific session.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **controlee** (const struct *controlee_parameters**) – Controlee to add.

2.3.44.1 Return

0 or positive value on success, negative value on error.

2.3.45 fira_helper_delete_controlee

int **fira_helper_delete_controlee**(struct fira_context *ctx, uint32_t session_handle, const struct *controlee_parameters* *controlee)

Delete one controlee from a specific session.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **controlee** (const struct *controlee_parameters**) – Controlee to delete.

2.3.45.1 Return

0 or positive value on success, negative value on error.

2.3.46 fira_helper_get_controlees

enum qerr **fira_helper_get_controlees**(struct fira_context *ctx, uint32_t session_handle, struct *controlees_parameters* *controlees)

Get controlees list.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **controlees** (struct *controlees_parameters**) – List of controlees to write.

2.3.46.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.47 fira_helper_get_controlees_count

enum qerr **fira_helper_get_controlees_count**(struct fira_context *ctx, uint32_t session_handle, int *count)

Get number of currently known controlees.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **count** (int*) – Number of controlees known.

2.3.47.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.48 fira_helper_data_message_send

enum qerr **fira_helper_data_message_send**(struct fira_context *ctx, uint32_t session_handle, const struct *data_message_content* *data_content)

[Not supported in QM33 SDK] Send data message.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **data_content** (const struct *data_message_content**) – Data message.

2.3.48.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.49 fira_helper_set_session_device_type

enum qerr **fira_helper_set_session_device_type**(struct fira_context *ctx, uint32_t session_handle, uint8_t device_type)

Sets the device type.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **device_type** (uint8_t) – 0 - CONTROLEE, 1 - CONTROLLER.

2.3.49.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.50 fira_helper_set_session_dl_tdoa_time_reference_anchor

enum qerr **fira_helper_set_session_dl_tdoa_time_reference_anchor**(struct fira_context *ctx, uint32_t session_handle, uint8_t global_time)

[Not supported in QM33 SDK] Set or reset the time reference anchor.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **global_time** (uint8_t) – 0 - DISABLE, 1 - Set DT-ANCHOR as global time reference and sets its cost metric to zero.

2.3.50.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.51 fira_helper_set_session_dl_tdoa_responder_tof

enum qerr **fira_helper_set_session_dl_tdoa_responder_tof**(struct fira_context *ctx, uint32_t session_handle, uint8_t responder_tof)

[Not supported in QM33 SDK] Include or not the responder tof.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **responder_tof** (uint8_t) – 0 - Do not include, 1 - include the estimated Responder ToF Result in a Response DTM.

2.3.51.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.52 fira_helper_set_session_dl_tdoa_ranging_method

enum qerr **fira_helper_set_session_dl_tdoa_ranging_method**(struct fira_context *ctx, uint32_t session_handle, uint8_t method)

[Not supported in QM33 SDK] Set dl-tdoa ranging method.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **method** (uint8_t) – 0 - SS-TWR, 1 - DS-TWR.

2.3.52.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.53 fira_helper_set_session_dl_tdoa_tx_timestamp_type

enum qerr **fira_helper_set_session_dl_tdoa_tx_timestamp_type**(struct fira_context *ctx, uint32_t session_handle, uint8_t type)

[Not supported in QM33 SDK] Configure tx timestamp type.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **type** (uint8_t) – Timestamp type.

2.3.53.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.54 fira_helper_set_session_dl_tdoa_tx_timestamp_len

enum qerr **fira_helper_set_session_dl_tdoa_tx_timestamp_len**(struct fira_context *ctx, uint32_t session_handle, uint8_t len)

[Not supported in QM33 SDK] Configure tx timestamp length.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **len** (uint8_t) – Timestamp length.

2.3.54.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.55 fira_helper_set_session_dl_tdoa_hop_count

enum qerr **fira_helper_set_session_dl_tdoa_hop_count**(struct fira_context *ctx, uint32_t session_handle, uint8_t count)

[Not supported in QM33 SDK] Set dl-tdoa hop count presence.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **count** (uint8_t) – 0 - DEACTIVATED, 1 - ACTIVATED.

2.3.55.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.56 fira_helper_set_session_dl_tdoa_anchor_cfo

enum qerr **fira_helper_set_session_dl_tdoa_anchor_cfo**(struct fira_context *ctx, uint32_t session_handle, uint8_t cfo)

[Not supported in QM33 SDK] Set dl-tdoa presence of cfo.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **cfo** (uint8_t) – 0 - DEACTIVATED, 1 - ACTIVATED.

2.3.56.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.57 fira_helper_set_session_dl_tdoa_anchor_location_presence

enum qerr **fira_helper_set_session_dl_tdoa_anchor_location_presence**(struct fira_context *ctx, uint32_t session_handle, uint8_t presence)

[Not supported in QM33 SDK] Set dl-tdoa presence of dt-anchor location.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **presence** (uint8_t) – presence of the information about DT-Anchor location in DTMs 0 - DEACTIVATED, 1 - ACTIVATED.

2.3.57.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.58 fira_helper_set_session_dl_tdoa_anchor_location

enum qerr **fira_helper_set_session_dl_tdoa_anchor_location**(struct fira_context *ctx, uint32_t session_handle, uint8_t len, uint8_t *data)

[Not supported in QM33 SDK] Set dl-tdoa dt-anchor location.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **len** (uint8_t) – Length of the array according to the location type.
- **data** (uint8_t*) – data array that represents location.

2.3.58.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.59 fira_helper_set_session_dl_tdoa_anchor_location_type

enum qerr **fira_helper_set_session_dl_tdoa_anchor_location_type**(struct fira_context *ctx, uint32_t session_handle, uint8_t type)

[Not supported in QM33 SDK] Set dl-tdoa type of dt-anchor location.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **type** (uint8_t) – Type of the DT-Anchor location format: 0 - WGS84, 1 - relative.

2.3.59.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.60 fira_helper_set_session_dl_tdoa_active_ranging_rounds

enum qerr **fira_helper_set_session_dl_tdoa_active_ranging_rounds**(struct fira_context *ctx, uint32_t session_handle, uint8_t rrounds)

[Not supported in QM33 SDK] Set dl-tdoa presence of ranging rounds.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **rrounds** (uint8_t) – 0 - deactivated, 1 - activated.

2.3.60.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.61 fira_helper_set_session_dl_tdoa_block_skipping

enum qerr **fira_helper_set_session_dl_tdoa_block_skipping**(struct fira_context *ctx, uint32_t session_handle, uint8_t number)

[Not supported in QM33 SDK] Set dl-tdoa number of blocks that shall be skipped between 2 active ranging blocks.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **number** (uint8_t) – Number of blocks to be skipped by the dt-tag.

2.3.61.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.62 fira_helper_set_session_report_psdus

enum qerr **fira_helper_set_session_report_psdus**(struct fira_context *ctx, uint32_t session_handle, uint8_t active)

[Not supported in QM33 SDK] Enable/disable psdus report.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **active** (uint8_t) – True to enable psdus are reported, false otherwise.

2.3.62.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.63 fira_helper_get_session_in_band_termination_attempt_count

enum qerr **fira_helper_get_session_in_band_termination_attempt_count**(struct fira_context *ctx, uint32_t session_handle, uint8_t *termination_count)

Get the in band termination attempt count.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **termination_count** (uint8_t*) – Termination_count.

2.3.63.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.64 fira_helper_get_session_dl_tdoa_time_reference_anchor

enum qerr **fira_helper_get_session_dl_tdoa_time_reference_anchor**(struct fira_context *ctx, uint32_t session_handle, uint8_t *global_time)

[Not supported in QM33 SDK] Get the time reference anchor.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **global_time** (uint8_t*) – Time reference anchor.

2.3.64.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.65 fira_helper_get_session_dl_tdoa_responder_tof

enum qerr **fira_helper_get_session_dl_tdoa_responder_tof**(struct fira_context *ctx, uint32_t session_handle, uint8_t *responder_tof)

[Not supported in QM33 SDK] Get the responder tof config.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **responder_tof** (uint8_t*) – Responder time of flight.

2.3.65.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.66 fira_helper_get_session_dl_tdoa_ranging_method

enum qerr **fira_helper_get_session_dl_tdoa_ranging_method**(struct fira_context *ctx, uint32_t session_handle, uint8_t *method)

[Not supported in QM33 SDK] Get dl-tdoa ranging method.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **method** (uint8_t*) – 0 - SS-TWR, 1 - DS-TWR.

2.3.66.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.67 fira_helper_get_session_dl_tdoa_tx_timestamp_type

enum qerr **fira_helper_get_session_dl_tdoa_tx_timestamp_type**(struct fira_context *ctx, uint32_t session_handle, uint8_t *type)

[Not supported in QM33 SDK] Get tx timestamp type.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **type** (uint8_t*) – Timestamp type.

2.3.67.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.68 fira_helper_get_session_dl_tdoa_tx_timestamp_len

enum qerr **fira_helper_get_session_dl_tdoa_tx_timestamp_len**(struct fira_context *ctx, uint32_t session_handle, uint8_t *len)

[Not supported in QM33 SDK] Get tx timestamp length.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **len** (uint8_t*) – Timestamp length.

2.3.68.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.69 fira_helper_get_session_dl_tdoa_hop_count

enum qerr **fira_helper_get_session_dl_tdoa_hop_count**(struct fira_context *ctx, uint32_t session_handle, uint8_t *count)

[Not supported in QM33 SDK] Get dl-tdoa hop count presence.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **count** (uint8_t*) – 0 - DEACTIVATED, 1 - ACTIVATED.

2.3.69.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.70 fira_helper_get_session_dl_tdoa_anchor_cfo

enum qerr **fira_helper_get_session_dl_tdoa_anchor_cfo**(struct fira_context *ctx, uint32_t session_handle, uint8_t *cfo)

[Not supported in QM33 SDK] Get dl-tdoa presence of cfo.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **cfo** (uint8_t*) – 0 - DEACTIVATED, 1 - ACTIVATED.

2.3.70.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.71 fira_helper_get_session_dl_tdoa_anchor_location_presence

enum qerr **fira_helper_get_session_dl_tdoa_anchor_location_presence**(struct fira_context *ctx, uint32_t session_handle, uint8_t *presence)

[Not supported in QM33 SDK] Get dl-tdoa presence of dt-anchor location.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **presence** (uint8_t*) – presence of the information about DT-Anchor location in DTMs 0 - DEACTIVATED, 1 - ACTIVATED.

2.3.71.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.72 fira_helper_get_session_dl_tdoa_anchor_location_type

enum qerr **fira_helper_get_session_dl_tdoa_anchor_location_type**(struct fira_context *ctx, uint32_t session_handle, uint8_t *type)

[Not supported in QM33 SDK] Get dl-tdoa type of dt-anchor location.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **type** (uint8_t*) – Type of the DT-Anchor location format: 0 - WGS84, 1 - RELATIVE.

2.3.72.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.73 fira_helper_get_session_dl_tdoa_anchor_location

enum qerr **fira_helper_get_session_dl_tdoa_anchor_location**(struct fira_context *ctx, uint32_t session_handle, uint8_t len, uint8_t *data)

[Not supported in QM33 SDK] Get dl-tdoa dt-anchor location.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **len** (uint8_t) – Length of the array according to the location type.
- **data** (uint8_t*) – data array that represents location.

2.3.73.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.74 fira_helper_get_session_dl_tdoa_active_ranging_rounds

enum qerr **fira_helper_get_session_dl_tdoa_active_ranging_rounds**(struct fira_context *ctx, uint32_t session_handle, uint8_t *rrounds)

[Not supported in QM33 SDK] Get dl-tdoa presence of ranging rounds.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **rrounds** (uint8_t*) – 0 - DEACTIVATED, 1 - ACTIVATED.

2.3.74.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.75 fira_helper_get_session_dl_tdoa_block_skipping

enum qerr **fira_helper_get_session_dl_tdoa_block_skipping**(struct fira_context *ctx, uint32_t session_handle, uint8_t *number)

[Not supported in QM33 SDK] Get dl-tdoa number of blocks that shall be skipped between 2 active ranging blocks.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **number** (uint8_t*) – Number of blocks to be skipped by the dt-tag.

2.3.75.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.76 fira_helper_get_session_report_psdus

enum qerr **fira_helper_get_session_report_psdus**(struct fira_context *ctx, uint32_t session_handle, uint8_t *active)

[Not supported in QM33 SDK] Get activation of psdus report.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **active** (uint8_t*) – True if psdus are reported, false otherwise

2.3.76.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.77 fira_helper_bool_to_ranging_round_control

uint8_t **fira_helper_bool_to_ranging_round_control**(bool result_report_phase, bool skip_ranging_control_phase)

get the ranging round control bitfield format.

Parameters

- **result_report_phase** (bool) – True if result report phase present.
- **skip_ranging_control_phase** (bool) – True if ranging control phase is skipped.

2.3.77.1 Return

ranging round control bitfield format.

2.3.78 fira_helper_set_session_ranging_round_usage

enum qerr **fira_helper_set_session_ranging_round_usage**(struct fira_context *ctx, uint32_t session_handle, uint8_t ranging_round_usage)

Sets ranging round usage.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **ranging_round_usage** (uint8_t) – See *enum fira_ranging_round_usage*.

2.3.78.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.79 fira_helper_set_session_device_role

enum qerr **fira_helper_set_session_device_role**(struct fira_context *ctx, uint32_t session_handle, uint8_t device_role)

Sets the device role

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **device_role** (uint8_t) – Role played by the device, accepted value are initiator for controller and responder for controlee.

2.3.79.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.80 fira_helper_set_session_sts_config

enum qerr **fira_helper_set_session_sts_config**(struct fira_context *ctx, uint32_t session_handle, uint8_t sts_config)

scrambled timestamp sequence configuration.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **sts_config** (uint8_t) – Possible values: 0x01: Static STS (default). 0x02: Dynamic STS. *[Not supported in QM33 SDK]* 0x04: RFU (Dynamic STS - Individual Key). *[Not supported in QM33 SDK]* 0x08: Provisioned STS. 0x10: RFU (Provisioned STS - Individual Key).

2.3.80.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.81 fira_helper_set_session_multi_node_mode

enum qerr **fira_helper_set_session_multi_node_mode**(struct fira_context *ctx, uint32_t session_handle, uint8_t multi_node_mode)

The multi-node mode used during a round.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **multi_node_mode** (uint8_t) –

- FIRA_MULTI_NODE_MODE_UNICAST,
- FIRA_MULTI_NODE_MODE_ONE_TO_MANY,
- **[NOT IMPLEMENTED]** FIRA_MULTI_NODE_MODE_MANY_TO_MANY,

2.3.81.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.82 fira_helper_set_session_short_address

enum qerr **fira_helper_set_session_short_address**(struct fira_context *ctx, uint32_t session_handle, uint16_t short_addr)

Sets short address.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **short_addr** (uint16_t) – Short_addr.

2.3.82.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.83 fira_helper_set_session_destination_short_addresses

enum qerr **fira_helper_set_session_destination_short_addresses**(struct fira_context *ctx, uint32_t session_handle, uint32_t n_dest_short_addr, uint16_t *dest_short_addr)

Sets destination short addresses.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **n_dest_short_addr** (uint32_t) – Number of destination short addresses.
- **dest_short_addr** (uint16_t*) – Array of destination short addresses.

2.3.83.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.84 fira_helper_set_session_time0_ns

enum qerr **fira_helper_set_session_time0_ns**(struct fira_context *ctx, uint32_t session_handle, uint64_t time0_ns)

Sets an absolute value of the initiation time [ns].

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **time0_ns** (uint64_t) – time0_ns.

2.3.84.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.85 fira_helper_set_session_slot_duration_rstu

enum qerr **fira_helper_set_session_slot_duration_rstu**(struct fira_context *ctx, uint32_t session_handle, uint32_t slot_duration_rstu)

Sets slot duration rstu.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **slot_duration_rstu** (uint32_t) – Slot_duration_rstu. - Duration of a slot in RSTU (1200RSTU=1ms)

2.3.85.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.86 fira_helper_set_session_round_duration_slots

enum qerr **fira_helper_set_session_round_duration_slots**(struct fira_context *ctx, uint32_t session_handle, uint32_t round_duration_slots)

Sets round duration slots.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **round_duration_slots** (uint32_t) – Number of slots per ranging round.

2.3.86.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.87 fira_helper_set_session_block_duration_ms

enum qerr **fira_helper_set_session_block_duration_ms**(struct fira_context *ctx, uint32_t session_handle, uint32_t block_duration_ms)

Sets block duration.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **block_duration_ms** (uint32_t) – Block size in unit of 1200 RSTU (same as ms).

2.3.87.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.88 fira_helper_set_session_time_base

enum qerr **fira_helper_set_session_time_base**(struct fira_context *ctx, uint32_t session_handle, const uint8_t *time_base_param)

[Not supported in QM33 SDK] Set session time base configuration.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **time_base_param** (const uint8_t*) – Session time base parameter array. Expected array size equals to FIRA_TIME_BASE_SIZE.

2.3.88.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.89 fira_helper_set_session_block_stride_length

enum qerr **fira_helper_set_session_block_stride_length**(struct fira_context *ctx, uint32_t session_handle, uint32_t block_stride_length)

Sets block stride length.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **block_stride_length** (uint32_t) – Number of blocks to stride.

2.3.89.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.90 fira_helper_set_session_round_hopping

enum qerr **fira_helper_set_session_round_hopping**(struct fira_context *ctx, uint32_t session_handle, uint8_t round_hopping)

Enables or disable round hopping

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **round_hopping** (uint8_t) – False - disabled, true - enabled

2.3.90.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.91 fira_helper_set_session_priority

enum qerr **fira_helper_set_session_priority**(struct fira_context *ctx, uint32_t session_handle, uint8_t priority)

sets the priority.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **priority** (uint8_t) – Priority of the session.

2.3.91.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.92 fira_helper_set_session_mac_address_mode

enum qerr **fira_helper_set_session_mac_address_mode**(struct fira_context *ctx, uint32_t session_handle, uint8_t mac_address_mode)

sets the MAC addressing mode.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **mac_address_mode** (uint8_t) – MAC addressing mode.

2.3.92.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.93 fira_helper_set_session_ranging_round_control

enum qerr **fira_helper_set_session_ranging_round_control**(struct fira_context *ctx, uint32_t session_handle, uint8_t ranging_round_control)

Set the ranging round control bitfield.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **ranging_round_control** (uint8_t) – Bitfield: - b0: ranging result report phase is disabled (0) or enabled (1) - b1: Control Message is sent in band (1) or not (0, not supported) - b2: Control Message is sent separately (0) or piggybacked to RIM (1)

2.3.93.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.94 fira_helper_set_session_schedule_mode

enum qerr **fira_helper_set_session_schedule_mode**(struct fira_context *ctx, uint32_t session_handle, uint8_t schedule_mode)

Sets schedule mode parameter.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **schedule_mode** (uint8_t) –
  - 0x00 - Contention-based ranging. *[Not supported in QM33 SDK]*
  - 0x01 - Time-scheduled ranging.
  - 0x02 - Hybrid-based ranging. *[Not supported in QM33 SDK]*

2.3.94.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.95 fira_helper_set_session_max_number_of_measurements

enum qerr **fira_helper_set_session_max_number_of_measurements**(struct fira_context *ctx, uint32_t session_handle, uint32_t max_number_of_measurements)

Sets the max number of measurements.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **max_number_of_measurements** (uint32_t) – Max_number_of_measurements.

2.3.95.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.96 fira_helper_set_session_max_rr_retry

enum qerr **fira_helper_set_session_max_rr_retry**(struct fira_context *ctx, uint32_t session_handle, uint32_t max_rr_retry)

Sets the max rr retry.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **max_rr_retry** (uint32_t) – Max_rr_retry. Number of failed ranging round attempts before stopping the session. The value zero disables the feature.

2.3.96.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.97 fira_helper_set_session_channel_number

enum qerr **fira_helper_set_session_channel_number**(struct fira_context *ctx, uint32_t session_handle, uint8_t channel_number)

Sets the channel number.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **channel_number** (uint8_t) – Channel_number.

2.3.97.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.98 fira_helper_set_session_preamble_code_index

enum qerr **fira_helper_set_session_preamble_code_index**(struct fira_context *ctx, uint32_t session_handle, uint8_t preamble_code_index)

Sets preamble code index.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **preamble_code_index** (uint8_t) – Preamble code index.

  Possible values:

  – 9-24: BPRF

  – 25-32: HPRF

2.3.98.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.99 fira_helper_set_session_rframe_config

enum qerr **fira_helper_set_session_rframe_config**(struct fira_context *ctx, uint32_t session_handle, uint8_t rframe_config)

Sets rframe_config.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **rframe_config** (uint8_t) – Ranging frame config.

2.3.99.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.100 fira_helper_set_session_preamble_duration

enum qerr **fira_helper_set_session_preamble_duration**(struct fira_context *ctx, uint32_t session_handle, uint8_t preamble_duration)

Sets preamble duration.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.

- **`preamble_duration`** (`uint8_t`) – 0x00: 32 symbols *[Not supported in QM33 SDK]* or 0x01: 64 symbols (default)

2.3.100.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.101 fira_helper_set_session_sfd_id

enum qerr **`fira_helper_set_session_sfd_id`**(struct fira_context *ctx, uint32_t session_handle, uint8_t sfd_id)

Sets sfd_id.

Parameters

- **`ctx`** (`struct fira_context*`) – Fira helper context.
- **`session_handle`** (`uint32_t`) – Session handle.
- **`sfd_id`** (`uint8_t`) – 0 or 2 in BPRF, 1-4 in HPRF *[Not supported in QM33 SDK]*

2.3.101.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.102 fira_helper_set_session_psdu_data_rate

enum qerr **`fira_helper_set_session_psdu_data_rate`**(struct fira_context *ctx, uint32_t session_handle, uint8_t psdu_data_rate)

Sets psdu data rate.

Parameters

- **`ctx`** (`struct fira_context*`) – Fira helper context.
- **`session_handle`** (`uint32_t`) – Session handle.
- **`psdu_data_rate`** (`uint8_t`) – Possible values: - 0: 6.81Mbps (default) - 1: 7.80 Mbps *[Not supported in QM33 SDK]* - 2: 27.2 Mbps *[Not supported in QM33 SDK]* - 3: 31.2 Mbps *[Not supported in QM33 SDK]*

2.3.102.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.103 fira_helper_set_session_sub_session_id

enum qerr **`fira_helper_set_session_sub_session_id`**(struct fira_context *ctx, uint32_t session_handle, uint32_t sub_session_id)

Sets controlee' sub-session id.

Parameters

- **`ctx`** (`struct fira_context*`) – Fira helper context.
- **`session_handle`** (`uint32_t`) – Session handle.

- **sub_session_id** (uint32_t) – Controlee' sub-session id used during Dynamic or Provisioned STS for Responder Specific Sub-session Key.

2.3.103.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.104 fira_helper_set_session_vendor_id

enum qerr **fira_helper_set_session_vendor_id**(struct fira_context *ctx, uint32_t session_handle, uint8_t vendor_id)

Sets Vendor ID.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **vendor_id** (uint8_t) – Vendor ID used for vUpper64.

2.3.104.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.105 fira_helper_set_session_static_sts_iv

enum qerr **fira_helper_set_session_static_sts_iv**(struct fira_context *ctx, uint32_t session_handle, uint8_t static_sts_iv)

Sets Static STS IV.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **static_sts_iv** (uint8_t) – Static STS IV used for vUpper64.

2.3.105.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.106 fira_helper_set_session_vupper64

enum qerr **fira_helper_set_session_vupper64**(struct fira_context *ctx, uint32_t session_handle, uint8_t vupper64)

Sets vupper64.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **vupper64** (uint8_t) – vUpper64 used during Static STS ranging.

2.3.106.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.107 fira_helper_set_session_key_rotation

enum qerr **fira_helper_set_session_key_rotation**(struct fira_context *ctx, uint32_t session_handle, uint8_t key_rotation)

Enable/disable key rotation.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **key_rotation** (uint8_t) – 0 to disable key rotation, 1 to enable it. Enable/disable key rotation during Dynamic *[Not supported in QM33 SDK]* or Provisioned STS ranging. If enable the period will be set with fira_helper_set_session_key_rotation_rate.

2.3.107.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.108 fira_helper_set_session_key_rotation_rate

enum qerr **fira_helper_set_session_key_rotation_rate**(struct fira_context *ctx, uint32_t session_handle, uint8_t key_rotation_rate)

Sets key rotation rate.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **key_rotation_rate** (uint8_t) – Defines n, with 2^n being the rotation rate of some keys used during Dynamic *[Not supported in QM33 SDK]* or Provisioned STS Ranging, n shall be in the range of 0<=n<=15.

2.3.108.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.109 fira_helper_set_session_mac_payload_encryption

enum qerr **fira_helper_set_session_mac_payload_encryption**(struct fira_context *ctx, uint32_t session_handle, uint8_t mac_payload_encryption)

Enable or disable encryption of payload data.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.

- **mac_payload_encryption** (uint8_t) – Status of mac payload encryption.

2.3.109.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.110 fira_helper_set_session_report_rssi

enum qerr **fira_helper_set_session_report_rssi**(struct fira_context *ctx, uint32_t session_handle, uint8_t report_rssi)

Sets the report rssi.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **report_rssi** (uint8_t) – Report_rssi false - no report, true report

2.3.110.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.111 fira_helper_bool_to_result_report_config

uint8_t **fira_helper_bool_to_result_report_config**(bool report_tof, bool report_aoa_azimuth, bool report_aoa_elevation, bool report_aoa_fom)

get the result report config bitfield format.

Parameters

- **report_tof** (bool) – True if time of flight must be reported.
- **report_aoa_azimuth** (bool) – True if azimuth's angle of arrival must be reported.
- **report_aoa_elevation** (bool) – True if elevation's angle of arrival must be reported.
- **report_aoa_fom** (bool) – True if aoa figure of merit must be reported.

2.3.111.1 Return

result report config bitfield format.

2.3.112 fira_helper_set_session_result_report_config

enum qerr **fira_helper_set_session_result_report_config**(struct fira_context *ctx, uint32_t session_handle, uint8_t result_report_config)

Enable/disable time of flight.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.

- **result_report_config** (uint8_t) – See enum fira_result_report_config.

2.3.112.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.113 fira_helper_set_session_link_layer_mode

enum qerr **fira_helper_set_session_link_layer_mode**(struct fira_context *ctx, uint32_t session_handle, uint8_t link_layer_mode)

Sets link layer mode.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **link_layer_mode** (uint8_t) – Link layer configuration: 0x00: Bypass mode. 0x01: Connection less. *[Not supported in QM33 SDK]*

2.3.113.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.114 fira_helper_set_session_data_repetition_count

enum qerr **fira_helper_set_session_data_repetition_count**(struct fira_context *ctx, uint32_t session_handle, uint8_t data_repetition_count)

[Not supported in QM33 SDK] Sets data repetition count.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **data_repetition_count** (uint8_t) – Number of times the current MDSDU shall be sent. 0x00: No repetition. 0x01 - 0xFE: Number of repetitions. 0xFF: Infinite number of times.

2.3.114.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.115 fira_helper_set_session_data_transfer_status_ntf_config

enum qerr **fira_helper_set_session_data_transfer_status_ntf_config**(struct fira_context *ctx, uint32_t session_handle, uint8_t config)

[Not supported in QM33 SDK] Sets config value for SESSION_DATA_TRANSFER_STATUS_NTF.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **config** (uint8_t) – Value to set. 0x00: Disable 0x01: Enable

2.3.115.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.116 fira_helper_set_session_mac_fcs_type

enum qerr **fira_helper_set_session_mac_fcs_type**(struct fira_context *ctx, uint32_t session_handle, uint8_t mac_fcs_type)

Sets the CRC type.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **mac_fcs_type** (uint8_t) – CRC type: 0x00: CRC 16. 0x01: CRC 32.

2.3.116.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.117 fira_helper_set_session_number_of_sts_segments

enum qerr **fira_helper_set_session_number_of_sts_segments**(struct fira_context *ctx, uint32_t session_handle, uint8_t number_of_sts_segments)

Sets the number of STS segments.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **number_of_sts_segments** (uint8_t) – Number of STS segments: 0x00: No STS Segments. 0x01: 1 STS Segment. 0x02: 2 STS Segments (HPRF only). *[Not supported in QM33 SDK]* 0x03: 3 STS Segments (HPRF only). *[Not supported in QM33 SDK]* 0x04: 4 STS Segments (HPRF only). *[Not supported in QM33 SDK]*

2.3.117.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.118 fira_helper_set_session_phr_data_rate

enum qerr **fira_helper_set_session_phr_data_rate**(struct fira_context *ctx, uint32_t session_handle, uint8_t phr_data_rate)

Sets the PHR data rate.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **phr_data_rate** (uint8_t) – PHR data rate: 0x00: 850 kbps. 0x01: 6.81 Mbps.

2.3.118.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.119 fira_helper_set_session_prf_mode

enum qerr **fira_helper_set_session_prf_mode**(struct fira_context *ctx, uint32_t session_handle, uint8_t prf_mode)

Sets prf mode.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **prf_mode** (uint8_t) – Prf_mode pulse repetition frequency. 0x00: 62.4 MHz PRF. BPRF mode (default) 0x01: 124.8 MHz PRF. HPRF mode. *[Not supported in QM33 SDK]* 0x02: 249.6 MHz PRF. HPRF mode with data rate 27.2 and 31.2 Mbps. *[Not supported in QM33 SDK]*

2.3.119.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.120 fira_helper_set_session_cap_size_min

enum qerr **fira_helper_set_session_cap_size_min**(struct fira_context *ctx, uint32_t session_handle, uint8_t cap_size_min)

[Not supported in QM33 SDK] Sets the cap size min.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **cap_size_min** (uint8_t) – Cap_size_min - default 5.

2.3.120.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.121 fira_helper_set_session_cap_size_max

enum qerr **fira_helper_set_session_cap_size_max**(struct fira_context *ctx, uint32_t session_handle, uint8_t cap_size_max)

[Not supported in QM33 SDK] Sets the cap size max.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **cap_size_max** (uint8_t) – Cap_size_max.

2.3.121.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.122 fira_helper_set_session_measurement_sequence

enum qerr **fira_helper_set_session_measurement_sequence**(struct fira_context *ctx, uint32_t session_handle, const struct *measurement_sequence* *meas_seq)

[Not supported in QM33 SDK] Sets the measurement sequence.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **meas_seq** (const struct *measurement_sequence**) – Sequence of measurement sequence steps, configures the Antenna Flexibility features.

2.3.122.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.123 fira_helper_set_session_enable_diagnostics

enum qerr **fira_helper_set_session_enable_diagnostics**(struct fira_context *ctx, uint32_t session_handle, uint8_t enable_diagnostics)

Enables diagnostics.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **enable_diagnostics** (uint8_t) – Enable_diagnostics 0 - no, 1 - yes.

2.3.123.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.124 fira_helper_set_session_diags_frame_reports_fields

enum qerr **fira_helper_set_session_diags_frame_reports_fields**(struct fira_context *ctx, uint32_t session_handle, uint32_t diags_frame_reports_fields)

Sets the diag frame fields.

Parameters

- **ctx** (`struct fira_context*`) – Fira helper context.
- **session_handle** (`uint32_t`) – Session handle.
- **diags_frame_reports_fields** (`uint32_t`) – Select the fields to activate in the frame reports stored in the diags. Applicable only when `enable_diagnostics` is set to true.

2.3.124.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.125 fira_helper_set_session_sts_length

enum qerr **fira_helper_set_session_sts_length**(struct fira_context *ctx, uint32_t session_handle, uint8_t sts_length)

Sets sts length.

Parameters

- **ctx** (`struct fira_context*`) – Fira helper context.
- **session_handle** (`uint32_t`) – Session handle.
- **sts_length** (`uint8_t`) – Values 0x00: 32 symbols 0x01: 64 symbols (default) 0x02: 128 symbols Values 0x03 to 0xFF: RFU

2.3.125.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.126 fira_helper_set_session_min_frames_per_rr

enum qerr **fira_helper_set_session_min_frames_per_rr**(struct fira_context *ctx, uint32_t session_handle, uint8_t min_frames_per_rr)

[Not supported in QM33 SDK] Sets min_frames_per_rr

Parameters

- **ctx** (`struct fira_context*`) – Fira helper context.
- **session_handle** (`uint32_t`) – Session handle.
- **min_frames_per_rr** (`uint8_t`) – Min_frames_per_rr

2.3.126.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.127 fira_helper_set_session_mtu_size

enum qerr **fira_helper_set_session_mtu_size**(struct fira_context *ctx, uint32_t session_handle, uint16_t mtu_size)

[Not supported in QM33 SDK] Sets mtu_size

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **mtu_size** (uint16_t) – Mtu_size, the value shall be restricted to the maximum possible MTU size of the given frame which includes MHR, Variable IE size and FCS size.

2.3.127.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.128 fira_helper_set_session_inter_frame_interval_ms

enum qerr **fira_helper_set_session_inter_frame_interval_ms**(struct fira_context *ctx, uint32_t session_handle, uint8_t inter_frame_interval_ms)

[Not supported in QM33 SDK] Sets inter_frame_interval_ms

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **inter_frame_interval_ms** (uint8_t) – Inter_frame_interval_ms

2.3.128.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.129 fira_helper_set_session_owr_aoa_measurement_ntf_period

enum qerr **fira_helper_set_session_owr_aoa_measurement_ntf_period**(struct fira_context *ctx, uint32_t session_handle, uint8_t owr_aoa_measurement_ntf_period)

[Not supported in QM33 SDK] Sets OWR for AoA measurement notification period.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.

- **owr_aoa_measurement_ntf_period** (uint8_t) – 0 - send on every frame, 1 - send once after MIN_FRAMES_PER_RR number of AoA measurements are aggregated.

2.3.129.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.130 fira_helper_set_session_session_info_ntf_config

enum qerr **fira_helper_set_session_session_info_ntf_config**(struct fira_context *ctx, uint32_t session_handle, uint8_t session_info_ntf_config)

[Not supported in QM33 SDK] Sets ntf config.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **session_info_ntf_config** (uint8_t) – Values; 0x00 = Disable session info notification (ntf) 0x01 = Enable session info notification (default) 0x02 = Enable session info ntf while inside proximity range 0x03 = Enable session info ntf while inside AoA upper and lower bounds 0x04 = Enable session info ntf while inside AoA upper and lower bounds as well as inside proximity range 0x05 = Enable session info ntf only when entering or leaving proximity range 0x06 = Enable session info ntf only when entering or leaving AoA upper and lower bounds 0x07 = Enable session info ntf only when entering or leaving AoA upper and lower bounds as well as entering or leaving proximity range.

2.3.130.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.131 fira_helper_set_session_near_proximity_config_cm

enum qerr **fira_helper_set_session_near_proximity_config_cm**(struct fira_context *ctx, uint32_t session_handle, uint32_t near_proximity_config_cm)

[Not supported in QM33 SDK] set proximity near cm.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **near_proximity_config_cm** (uint32_t) – Range_data_ntf_proximity_near_cm. prerequisites: Applicable when session_info_ntf_config is set to 0x02, 0x04, 0x05 or 0x07. Should be less than or equal to far_proximity_config value.

2.3.131.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.132 fira_helper_set_session_far_proximity_config_cm

enum qerr **fira_helper_set_session_far_proximity_config_cm**(struct fira_context *ctx, uint32_t session_handle, uint32_t far_proximity_config_cm)

[Not supported in QM33 SDK] Sets far_proximity_config_cm. prerequisites, Applicable when session_info_ntf_config is set to 0x02, 0x04, 0x05 or 0x07. Should be greater than or equal to near_proximity_config value.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **far_proximity_config_cm** (uint32_t) – Range_data_ntf_proximity_far_cm.

2.3.132.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.133 fira_helper_get_session_device_type

enum qerr **fira_helper_get_session_device_type**(struct fira_context *ctx, uint32_t session_handle, uint8_t *device_type)

Gets the device type.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **device_type** (uint8_t*) – Variable to store the value.

2.3.133.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.134 fira_helper_get_session_ranging_round_usage

enum qerr **fira_helper_get_session_ranging_round_usage**(struct fira_context *ctx, uint32_t session_handle, uint8_t *ranging_round_usage)

Gets the ranging round usage.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **ranging_round_usage** (uint8_t*) – Variable to store the value.

2.3.134.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.135 fira_helper_get_session_device_role

enum qerr **fira_helper_get_session_device_role**(struct fira_context *ctx, uint32_t session_handle, uint8_t *device_role)

Gets device role.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **device_role** (uint8_t*) – Device_role.

2.3.135.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.136 fira_helper_get_session_sts_config

enum qerr **fira_helper_get_session_sts_config**(struct fira_context *ctx, uint32_t session_handle, uint8_t *sts_config)

Gets the sts config.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **sts_config** (uint8_t*) – Variable to store the value.

2.3.136.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.137 fira_helper_get_session_multi_node_mode

enum qerr **fira_helper_get_session_multi_node_mode**(struct fira_context *ctx, uint32_t session_handle, uint8_t *multi_node_mode)

Gets the multi node mode.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **multi_node_mode** (uint8_t*) – Variable to store the value.

2.3.137.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.138 fira_helper_get_session_short_address

enum qerr **fira_helper_get_session_short_address**(struct fira_context *ctx, uint32_t session_handle, uint16_t *short_addr)

Gets the short address.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **short_addr** (uint16_t*) – Variable to store the value.

2.3.138.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.139 fira_helper_get_session_destination_short_addresses

enum qerr **fira_helper_get_session_destination_short_addresses**(struct fira_context *ctx, uint32_t session_handle, uint32_t *n_dest_short_addr, uint16_t *dest_short_addr)

Gets the destination short addresses.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **n_dest_short_addr** (uint32_t*) – Number of destination short addresses.
- **dest_short_addr** (uint16_t*) – Array of destination short addresses.

2.3.139.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.140 fira_helper_get_session_time0_ns

enum qerr **fira_helper_get_session_time0_ns**(struct fira_context *ctx, uint32_t session_handle, uint64_t *time0_ns)

Gets an absolute value of the initiation time [ns].

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **time0_ns** (uint64_t*) – Variable to store the value.

2.3.140.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.141 fira_helper_get_session_slot_duration_rstu

enum qerr **fira_helper_get_session_slot_duration_rstu**(struct fira_context *ctx, uint32_t session_handle, uint32_t *slot_duration_rstu)

Gets the slot duration rstu.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **slot_duration_rstu** (uint32_t*) – Variable to store the value.

2.3.141.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.142 fira_helper_get_session_round_duration_slots

enum qerr **fira_helper_get_session_round_duration_slots**(struct fira_context *ctx, uint32_t session_handle, uint32_t *round_duration_slots)

Gets the number of round duration slots

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **round_duration_slots** (uint32_t*) – Variable to store the value.

2.3.142.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.143 fira_helper_get_session_block_duration_ms

enum qerr **fira_helper_get_session_block_duration_ms**(struct fira_context *ctx, uint32_t session_handle, uint32_t *block_duration_ms)

Gets the block duration.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **block_duration_ms** (uint32_t*) – Variable to store the value. (Block size in unit of 1200 RSTU (same as ms))

2.3.143.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.144 fira_helper_get_session_block_stride_length

enum qerr **fira_helper_get_session_block_stride_length**(struct fira_context *ctx, uint32_t session_handle, uint32_t *block_stride_length)

Gets the block stride length.

Parameters

- **ctx** (`struct fira_context*`) – Fira helper context.
- **session_handle** (`uint32_t`) – Session handle.
- **block_stride_length** (`uint32_t*`) – Variable to store the number of blocks to stride.

2.3.144.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.145 fira_helper_get_session_round_hopping

enum qerr **fira_helper_get_session_round_hopping**(struct fira_context *ctx, uint32_t session_handle, uint8_t *round_hopping)

Gets the round hopping 0 - disabled, 1 enabled.

Parameters

- **ctx** (`struct fira_context*`) – Fira helper context.
- **session_handle** (`uint32_t`) – Session handle.
- **round_hopping** (`uint8_t*`) – Variable to store the value.

2.3.145.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.146 fira_helper_get_session_priority

enum qerr **fira_helper_get_session_priority**(struct fira_context *ctx, uint32_t session_handle, uint8_t *priority)

Gets the priority.

Parameters

- **ctx** (`struct fira_context*`) – Fira helper context.
- **session_handle** (`uint32_t`) – Session handle.
- **priority** (`uint8_t*`) – Variable to store the value.

2.3.146.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.147 fira_helper_get_session_mac_address_mode

enum qerr **fira_helper_get_session_mac_address_mode**(struct fira_context *ctx, uint32_t session_handle, uint8_t *mac_address_mode)

Gets the MAC addressing mode.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **mac_address_mode** (uint8_t*) – Variable to store the value.

2.3.147.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.148 fira_helper_get_session_ranging_round_control

enum qerr **fira_helper_get_session_ranging_round_control**(struct fira_context *ctx, uint32_t session_handle, uint8_t *ranging_round_control)

Gets the ranging round control.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **ranging_round_control** (uint8_t*) – Storage variable, where ranging round control is: - b0: ranging result report phase is disabled (0) or enabled (1) - b1: Control Message is sent in band (1) or not (0) - b2: Control Message is sent separately (0) or piggybacked to RIM (1) - b3-b6: RFUs, must be set to 0. - b7: MRM is sent from the initiator (0) or from the responder (1)

2.3.148.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.149 fira_helper_get_session_schedule_mode

enum qerr **fira_helper_get_session_schedule_mode**(struct fira_context *ctx, uint32_t session_handle, uint8_t *schedule_mode)

Gets schedule mode parameter.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.

- **schedule_mode** (uint8_t*) – Variable to store the value. - 0x00 - Contention-based ranging. *[Not supported in QM33 SDK]* - 0x01 - Time-scheduled ranging. - 0x02 - Hybrid-based ranging. *[Not supported in QM33 SDK]*

2.3.149.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.150 fira_helper_get_session_max_number_of_measurements

enum qerr **fira_helper_get_session_max_number_of_measurements**(struct fira_context *ctx, uint32_t session_handle, uint32_t *max_number_of_measurements)

Gets the number of measurements.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **max_number_of_measurements** (uint32_t*) – Variable to store the value.

2.3.150.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.151 fira_helper_get_session_max_rr_retry

enum qerr **fira_helper_get_session_max_rr_retry**(struct fira_context *ctx, uint32_t session_handle, uint32_t *max_rr_retry)

Gets the maximum ranging rounds retry.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **max_rr_retry** (uint32_t*) – Max_rr_retry. Variable to store the value. Number of failed ranging round attempts before stopping the session. The value zero disables the feature.

2.3.151.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.152 fira_helper_get_session_channel_number

enum qerr **fira_helper_get_session_channel_number**(struct fira_context *ctx, uint32_t session_handle, uint8_t *channel_number)

Gets the channel used in this session.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **channel_number** (uint8_t*) – Variable to store the value.

2.3.152.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.153 fira_helper_get_session_preamble_code_index

enum qerr **fira_helper_get_session_preamble_code_index**(struct fira_context *ctx, uint32_t session_handle, uint8_t *preamble_code_index)

Gets preamble code index.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **preamble_code_index** (uint8_t*) – Variable to store the value, possible values: - 9-24: BPRF - 25-32: HPRF

2.3.153.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.154 fira_helper_get_session_rframe_config

enum qerr **fira_helper_get_session_rframe_config**(struct fira_context *ctx, uint32_t session_handle, uint8_t *rframe_config)

Gets the rframe_config.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **rframe_config** (uint8_t*) – Variable to store the value.

2.3.154.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.155 fira_helper_get_session_preamble_duration

enum qerr **fira_helper_get_session_preamble_duration**(struct fira_context *ctx, uint32_t session_handle, uint8_t *preamble_duration)

Gets the preamble duration.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **preamble_duration** (uint8_t*) – Variable to store the value. 0x00: 32 symbols *[Not supported in QM33 SDK]* or 0x01: 64 symbols (default)a.

2.3.155.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.156 fira_helper_get_session_sfd_id

enum qerr **fira_helper_get_session_sfd_id**(struct fira_context *ctx, uint32_t session_handle, uint8_t *sfd_id)

Gets sfd_id.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **sfd_id** (uint8_t*) – Sfd_id. Variable to store the value. possible values 0 or 2 in BPRF, 1-4 in HPRF *[Not supported in QM33 SDK]*

2.3.156.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.157 fira_helper_get_session_psdu_data_rate

enum qerr **fira_helper_get_session_psdu_data_rate**(struct fira_context *ctx, uint32_t session_handle, uint8_t *psdu_data_rate)

Gets the psdu data rate.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **psdu_data_rate** (uint8_t*) – Psdu_data_rate.

2.3.157.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.158 fira_helper_get_session_sub_session_id

enum qerr **fira_helper_get_session_sub_session_id**(struct fira_context *ctx, uint32_t session_handle, uint32_t *sub_session_id)

Gets controlee' sub-session id.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **sub_session_id** (uint32_t*) – Controlee' sub-session id used during Dynamic or Provisioned STS for Responder Specific Sub-session Key.

2.3.158.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.159 fira_helper_get_session_vendor_id

enum qerr **fira_helper_get_session_vendor_id**(struct fira_context *ctx, uint32_t session_handle, uint8_t vendor_id)

Gets the Vendor ID.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **vendor_id** (uint8_t) – Vendor ID.

2.3.159.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.160 fira_helper_get_session_static_sts_iv

enum qerr **fira_helper_get_session_static_sts_iv**(struct fira_context *ctx, uint32_t session_handle, uint8_t static_sts_iv)

Gets the Static STS IV.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **static_sts_iv** (uint8_t) – Static STS IV.

2.3.160.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.161 fira_helper_get_session_vupper64

enum qerr **fira_helper_get_session_vupper64**(struct fira_context *ctx, uint32_t session_handle, uint8_t vupper64)

Gets the vupper.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **vupper64** (uint8_t) – vupper64.

2.3.161.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.162 fira_helper_get_session_key_rotation

enum qerr **fira_helper_get_session_key_rotation**(struct fira_context *ctx, uint32_t session_handle, uint8_t *key_rotation)

Gets the key rotation.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **key_rotation** (uint8_t*) – False - no rotation, true rotation

2.3.162.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.163 fira_helper_get_session_key_rotation_rate

enum qerr **fira_helper_get_session_key_rotation_rate**(struct fira_context *ctx, uint32_t session_handle, uint8_t *key_rotation_rate)

Gets the key rotation rate.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **key_rotation_rate** (uint8_t*) – Value to store the variable which. defines n, with 2^n being the rotation rate of some keys used during Dynamic *[Not supported in QM33 SDK]* or Provisioned STS Ranging, n shall be in the range of 0<=n<=15.

2.3.163.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.164 fira_helper_get_session_mac_payload_encryption

enum qerr **fira_helper_get_session_mac_payload_encryption**(struct fira_context *ctx, uint32_t session_handle, uint8_t *mac_payload_encryption)

Gets the status of encryption of payload data.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **mac_payload_encryption** (uint8_t*) – Status of mac payload encryption.

2.3.164.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.165 fira_helper_get_session_report_rssi

enum qerr **fira_helper_get_session_report_rssi**(struct fira_context *ctx, uint32_t session_handle, uint8_t *report_rssi)

Gets rssi report.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **report_rssi** (uint8_t*) – Variable to store: false no report, true report.

2.3.165.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.166 fira_helper_get_session_result_report_config

enum qerr **fira_helper_get_session_result_report_config**(struct fira_context *ctx, uint32_t session_handle, uint8_t *result_report_config)

Gets the report tof

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **result_report_config** (uint8_t*) – Variable to store: false no report, true report.

2.3.166.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.167 fira_helper_get_session_data_vendor_oui

enum qerr **fira_helper_get_session_data_vendor_oui**(struct fira_context *ctx, uint32_t session_handle, uint32_t *data_vendor_oui)

gets the data vendor own unique id.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **data_vendor_oui** (uint32_t*) – Variable to store: false no report, true report.

2.3.167.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.168 fira_helper_get_session_link_layer_mode

enum qerr **fira_helper_get_session_link_layer_mode**(struct fira_context *ctx, uint32_t session_handle, uint8_t *link_layer_mode)

Gets link layer mode.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **link_layer_mode** (uint8_t*) – Link layer mode.

2.3.168.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.169 fira_helper_get_session_data_repetition_count

enum qerr **fira_helper_get_session_data_repetition_count**(struct fira_context *ctx, uint32_t session_handle, uint8_t *data_repetition_count)

[Not supported in QM33 SDK] Gets data repetition count.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **data_repetition_count** (uint8_t*) – Gets number of times each MDSDU shall be repeated.

2.3.169.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.170 fira_helper_get_session_data_transfer_status_ntf_config

enum qerr **fira_helper_get_session_data_transfer_status_ntf_config**(struct fira_context *ctx, uint32_t session_handle, uint8_t *config)

[Not supported in QM33 SDK] Gets config value for SESSION_DATA_TRANSFER_STATUS_NTF.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **config** (uint8_t*) – Gets config value for SESSION_DATA_TRANSFER_STATUS_NTF

2.3.170.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.171 fira_helper_get_session_time_base

enum qerr **fira_helper_get_session_time_base**(struct fira_context *ctx, uint32_t session_handle, uint8_t *time_base_param)

[Not supported in QM33 SDK] Get session time base configuration.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **time_base_param** (uint8_t*) – Gets session time base parameter array. Array size equals to FIRA_TIME_BASE_SIZE.

2.3.171.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.172 fira_helper_get_session_mac_fcs_type

enum qerr **fira_helper_get_session_mac_fcs_type**(struct fira_context *ctx, uint32_t session_handle, uint8_t *mac_fcs_type)

Gets CRC type.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **mac_fcs_type** (uint8_t*) – CRC type: 0x00: CRC 16. 0x01: CRC 32.

2.3.172.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.173 fira_helper_get_session_number_of_sts_segments

enum qerr **fira_helper_get_session_number_of_sts_segments**(struct fira_context *ctx, uint32_t session_handle, uint8_t *number_of_sts_segments)

Gets the number of STS segments.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **number_of_sts_segments** (uint8_t*) – Number of STS segments: 0x00: No STS Segments. 0x01: 1 STS Segment. 0x02: 2 STS Segments (HPRF only). *[Not supported in QM33 SDK]* 0x03: 3 STS Segments (HPRF only). *[Not supported in QM33 SDK]* 0x04: 4 STS Segments (HPRF only). *[Not supported in QM33 SDK]*

2.3.173.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.174 fira_helper_get_session_phr_data_rate

enum qerr **fira_helper_get_session_phr_data_rate**(struct fira_context *ctx, uint32_t session_handle, uint8_t *phr_data_rate)

Gets the PHR data rate.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **phr_data_rate** (uint8_t*) – PHR data rate: 0x00: 850 kbps. 0x01: 6.81 Mbps.

2.3.174.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.175 fira_helper_get_session_prf_mode

enum qerr **fira_helper_get_session_prf_mode**(struct fira_context *ctx, uint32_t session_handle, uint8_t *prf_mode)

gets the prf mode.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.

- **prf_mode** (uint8_t*) – Prf_mode. pulse repetition frequency Variable to store the value.

2.3.175.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.176 fira_helper_get_session_cap_size_min

enum qerr **fira_helper_get_session_cap_size_min**(struct fira_context *ctx, uint32_t session_handle, uint8_t *cap_size_min)

[Not supported in QM33 SDK] Gets cap size min.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **cap_size_min** (uint8_t*) – Variable to sore the value

2.3.176.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.177 fira_helper_get_session_cap_size_max

enum qerr **fira_helper_get_session_cap_size_max**(struct fira_context *ctx, uint32_t session_handle, uint8_t *cap_size_max)

[Not supported in QM33 SDK] Get cap size max.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **cap_size_max** (uint8_t*) – Variable to sore the value.

2.3.177.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.178 fira_helper_get_session_enable_diagnostics

enum qerr **fira_helper_get_session_enable_diagnostics**(struct fira_context *ctx, uint32_t session_handle, uint8_t *enable_diagnostics)

Enables diagnostics.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **enable_diagnostics** (uint8_t*) – Enable_diagnostics 0 - no, 1 - yes.

2.3.178.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.179 fira_helper_get_session_diags_frame_reports_fields

enum qerr **fira_helper_get_session_diags_frame_reports_fields**(struct fira_context *ctx, uint32_t session_handle, uint32_t *diags_frame_reports_fields)

Gets the diag frame fields.

Parameters

- **ctx** (`struct fira_context*`) – Fira helper context.
- **session_handle** (`uint32_t`) – Session handle.
- **diags_frame_reports_fields** (`uint32_t*`) – Select the fields to activate in the frame reports stored in the diags. Applicable only when `enable_diagnostics` is set to true.

2.3.179.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.180 fira_helper_get_session_measurement_sequence

enum qerr **fira_helper_get_session_measurement_sequence**(struct fira_context *ctx, uint32_t session_handle, struct *measurement_sequence* *meas_seq)

[Not supported in QM33 SDK] Gets the measurement sequence.

Parameters

- **ctx** (`struct fira_context*`) – Fira helper context.
- **session_handle** (`uint32_t`) – Session handle.
- **meas_seq** (`struct measurement_sequence*`) – Variable to store the measurement sequence

2.3.180.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.181 fira_helper_get_session_sts_length

enum qerr **fira_helper_get_session_sts_length**(struct fira_context *ctx, uint32_t session_handle, uint8_t *sts_length)

gets sts length.

Parameters

- **ctx** (`struct fira_context*`) – Fira helper context.
- **session_handle** (`uint32_t`) – Session handle.
- **sts_length** (`uint8_t*`) – Variable to store the value. 0x00: 32 symbols 0x01: 64 symbols (default) 0x02: 128 symbols Values 0x03 to 0xFF: RFU

2.3.181.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.182 fira_helper_get_session_min_frames_per_rr

enum qerr **fira_helper_get_session_min_frames_per_rr**(struct fira_context *ctx, uint32_t session_handle, uint8_t *min_frames_per_rr)

[Not supported in QM33 SDK] Gets min_frames_per_rr

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **min_frames_per_rr** (uint8_t*) – Min_frames_per_rr

2.3.182.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.183 fira_helper_get_session_mtu_size

enum qerr **fira_helper_get_session_mtu_size**(struct fira_context *ctx, uint32_t session_handle, uint16_t *mtu_size)

[Not supported in QM33 SDK] Gets mtu_size

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **mtu_size** (uint16_t*) – Mtu_size, the value shall be restricted to the maximum possible MTU size of the given frame which includes MHR, Variable IE size and FCS size.

2.3.183.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.184 fira_helper_get_session_inter_frame_interval_ms

enum qerr **fira_helper_get_session_inter_frame_interval_ms**(struct fira_context *ctx, uint32_t session_handle, uint8_t *inter_frame_interval_ms)

[Not supported in QM33 SDK] Gets inter_frame_interval_ms

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **inter_frame_interval_ms** (uint8_t*) – Inter_frame_interval_ms

2.3.184.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.185 fira_helper_get_session_owr_aoa_measurement_ntf_period

enum qerr **fira_helper_get_session_owr_aoa_measurement_ntf_period**(struct fira_context *ctx, uint32_t session_handle, uint8_t *owr_aoa_measurement_ntf_period)

[Not supported in QM33 SDK] Gets OWR for AoA measurement notification period.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **owr_aoa_measurement_ntf_period** (uint8_t*) – 0 - send on every frame, 1 - send once after MIN_FRAMES_PER_RR number of AoA measurements are aggregated.

2.3.185.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.186 fira_helper_get_session_session_info_ntf_config

enum qerr **fira_helper_get_session_session_info_ntf_config**(struct fira_context *ctx, uint32_t session_handle, uint8_t *session_info_ntf_config)

[Not supported in QM33 SDK] Gets range notification.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **session_info_ntf_config** (uint8_t*) – Variable to store the value: 0x00 = Disable session info notification (ntf). 0x01 = Enable session info notification (default). 0x02 = Enable session info ntf while inside proximity range. 0x03 = Enable session info ntf while inside AoA upper and lower bounds. 0x04 = Enable session info ntf while inside AoA upper and lower bounds as well as inside proximity range. 0x05 = Enable session info ntf only when entering or leaving proximity range 0x06 = Enable session info ntf only when entering or leaving AoA upper and lower bounds. 0x07 = Enable session info ntf only when entering or leaving AoA upper and lower bounds as well as entering or leaving proximity range.

2.3.186.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.187 fira_helper_get_session_near_proximity_config_cm

enum qerr **fira_helper_get_session_near_proximity_config_cm**(struct fira_context *ctx, uint32_t session_handle, uint32_t *near_proximity_config_cm)

[Not supported in QM33 SDK] Gets ntf proximity near.

Parameters

- **ctx** (`struct fira_context*`) – Fira helper context.
- **session_handle** (`uint32_t`) – Session handle.
- **near_proximity_config_cm** (`uint32_t*`) – Variable to store the value.

2.3.187.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.188 fira_helper_get_session_far_proximity_config_cm

enum qerr **fira_helper_get_session_far_proximity_config_cm**(struct fira_context *ctx, uint32_t session_handle, uint32_t *far_proximity_config_cm)

[Not supported in QM33 SDK] Gets ntf proximity far.

Parameters

- **ctx** (`struct fira_context*`) – Fira helper context.
- **session_handle** (`uint32_t`) – Session handle.
- **far_proximity_config_cm** (`uint32_t*`) – Variable to store the value.

2.3.188.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.189 fira_helper_set_session_lower_aoa_bound_config_azimuth_2pi

enum qerr **fira_helper_set_session_lower_aoa_bound_config_azimuth_2pi**(struct fira_context *ctx, uint32_t session_handle, const int32_t lower_aoa_bound_config_azimuth_2pi)

[Not supported in QM33 SDK] Sets ntf lower bound aoa azimuth.

Parameters

- **ctx** (`struct fira_context*`) – Fira helper context.
- **session_handle** (`uint32_t`) – Session handle.

- **lower_aoa_bound_config_azimuth_2pi** (const int32_t) – Lower bound in rad_2pi for AOA azimuth, applicable when session_info_ntf_config is set to 0x03, 0x04, 0x06 or 0x07. It is a signed value (rad_2pi). Allowed values range from -180° to +180°. should be less than or equal to SESSION_INFO_NTF_UPPER_BOUND_AOA_AZIMUTH value. (default = -180).

2.3.189.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.190 fira_helper_get_session_lower_aoa_bound_config_azimuth_2pi

enum qerr **fira_helper_get_session_lower_aoa_bound_config_azimuth_2pi**(struct fira_context *ctx, uint32_t session_handle, int32_t *lower_aoa_bound_config_azimuth_2pi)

[Not supported in QM33 SDK] Gets ntf lower bound aoa azimuth.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **lower_aoa_bound_config_azimuth_2pi** (int32_t*) – Lower bound in rad_2pi for AOA azimuth, applicable when session_info_ntf_config is set to 0x03, 0x04, 0x06 or 0x07. It is a signed value (rad_2pi). Allowed values range from -180° to +180°. should be less than or equal to SESSION_INFO_NTF_UPPER_BOUND_AOA_AZIMUTH value. (default = -180).

2.3.190.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.191 fira_helper_set_session_upper_aoa_bound_config_azimuth_2pi

enum qerr **fira_helper_set_session_upper_aoa_bound_config_azimuth_2pi**(struct fira_context *ctx, uint32_t session_handle, const int32_t data_ntf_upper_bound_aoa_azimuth_2pi)

[Not supported in QM33 SDK] Sets ntf upper bound aoa azimuth.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **data_ntf_upper_bound_aoa_azimuth_2pi** (const int32_t) – Upper bound in rad_2pi for AOA azimuth, applicable when session_info_ntf_config is set to 0x03, 0x04, 0x06 or 0x07. It is a signed value (rad_2pi). Allowed values range from -180° to +180°. Should be greater than or equal to SESSION_INFO_NTF_LOWER_BOUND_AOA_AZIMUTH value. (default = +180).

2.3.191.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.192 fira_helper_get_session_upper_aoa_bound_config_azimuth_2pi

enum qerr **fira_helper_get_session_upper_aoa_bound_config_azimuth_2pi**(struct fira_context *ctx, uint32_t session_handle, int32_t *data_ntf_upper_bound_aoa_azimuth_2pi)

[Not supported in QM33 SDK] Gets ntf upper bound aoa azimuth.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **data_ntf_upper_bound_aoa_azimuth_2pi** (int32_t*) – Upper bound in rad_2pi for AOA azimuth, applicable when session_info_ntf_config is set to 0x03, 0x04, 0x06 or 0x07. It is a signed value (rad_2pi). Allowed values range from -180° to +180°. Should be greater than or equal to SESSION_INFO_NTF_LOWER_BOUND_AOA_AZIMUTH value. (default = +180).

2.3.192.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.193 fira_helper_set_session_lower_aoa_bound_config_elevation_2pi

enum qerr **fira_helper_set_session_lower_aoa_bound_config_elevation_2pi**(struct fira_context *ctx, uint32_t session_handle, const int16_t data_ntf_lower_bound_aoa_elevation_2p

[Not supported in QM33 SDK] Sets ntf lower bound aoa elevation.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **data_ntf_lower_bound_aoa_elevation_2pi** (const int16_t) – Lower bound in rad_2pi for AOA elevation, applicable when session_info_ntf_config is set to 0x03, 0x04, 0x06 or 0x07. It is a signed value (rad_2pi). Allowed values range from -90° to +90°. Should be less than or equal to SESSION_INFO_NTF_UPPER_BOUND_AOA_ELEVATION value. (default = -90).

2.3.193.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.194 fira_helper_get_session_lower_aoa_bound_config_elevation_2pi

enum qerr **fira_helper_get_session_lower_aoa_bound_config_elevation_2pi**(struct fira_context *ctx, uint32_t session_handle, int16_t *data_ntf_lower_bound_aoa_elevation_2p

[Not supported in QM33 SDK] Gets ntf lower bound aoa elevation.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **data_ntf_lower_bound_aoa_elevation_2pi** (int16_t*) – Lower bound in rad_2pi for AOA elevation, applicable when session_info_ntf_config is set to 0x03, 0x04, 0x06 or 0x07. It is a signed value (rad_2pi). Allowed values range from -90° to +90°. Should be less than or equal to SESSION_INFO_NTF_UPPER_BOUND_AOA_ELEVATION value. (default = -90).

2.3.194.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.195 fira_helper_set_session_upper_aoa_bound_config_elevation_2pi

enum qerr **fira_helper_set_session_upper_aoa_bound_config_elevation_2pi**(struct fira_context *ctx, uint32_t session_handle, const int16_t data_ntf_upper_bound_aoa_elevation_2p

[Not supported in QM33 SDK] Sets ntf upper bound aoa elevation.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **data_ntf_upper_bound_aoa_elevation_2pi** (const int16_t) – Upper bound in rad_2pi for AOA elevation, applicable when session_info_ntf_config is set to 0x03, 0x04, 0x06 or 0x07. It is a signed value (rad_2pi). Allowed values range from -90° to +90°. Should be greater than or equal to SESSION_INFO_NTF_LOWER_BOUND_AOA_ELEVATION value. (default = +90).

2.3.195.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.196 fira_helper_get_session_upper_aoa_bound_config_elevation_2pi

enum qerr **fira_helper_get_session_upper_aoa_bound_config_elevation_2pi**(struct fira_context *ctx, uint32_t session_handle, int16_t *data_ntf_upper_bound_aoa_elevation_2

[Not supported in QM33 SDK] Gets ntf upper bound aoa elevation.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **data_ntf_upper_bound_aoa_elevation_2pi** (int16_t*) – Upper bound in rad_2pi for AOA elevation, applicable when session_info_ntf_config is set to 0x03, 0x04, 0x06 or 0x07. It is a signed value (rad_2pi). Allowed values range from -90° to +90°. Should be greater than or equal to SESSION_INFO_NTF_LOWER_BOUND_AOA_ELEVATION value. (default = +90).

2.3.196.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.197 fira_helper_set_session_key

enum qerr **fira_helper_set_session_key**(struct fira_context *ctx, uint32_t session_handle, const void *key, uint8_t size)

Sets this key for the session.

Parameters

- **ctx** (struct fira_context*) – FiRa helper context.
- **session_handle** (uint32_t) – Handle of the session to modify.
- **key** (const void*) – Pointer to the session key
- **size** (uint8_t) – Length of the session key, can be 128 or 256 bits.

2.3.197.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.198 fira_helper_set_sub_session_key

enum qerr **fira_helper_set_sub_session_key**(struct fira_context *ctx, uint32_t session_handle, const void *key, uint8_t size)

Sets key for the current controlee sub-session.

Parameters

- **ctx** (struct fira_context*) – FiRa helper context.
- **session_handle** (uint32_t) – Handle of the session to modify.
- **key** (const void*) – Pointer to the sub-session key

- **size** (uint8_t) – Length of the sub-session key, can be 128 or 256 bits.

2.3.198.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.199 fira_helper_set_session_in_band_termination_attempt_count

enum qerr **fira_helper_set_session_in_band_termination_attempt_count**(struct fira_context *ctx, uint32_t session_handle, uint8_t termination_count)

Sets in band termination attemp count.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **termination_count** (uint8_t) – In band termination attempt count to set

2.3.199.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.200 fira_helper_update_dt_anchor_ranging_rounds

enum qerr **fira_helper_update_dt_anchor_ranging_rounds**(struct fira_context *ctx, uint32_t session_handle, struct *update_dt_anchor_ranging_rounds_cmd* *cmd, struct *update_dt_anchor_ranging_rounds_rsp* *rsp)

[Not supported in QM33 SDK] Configure ranging rounds for DT-Anchor.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **cmd** (struct *update_dt_anchor_ranging_rounds_cmd**) – Command containing configuration parameters of the ranging rounds.
- **rsp** (struct *update_dt_anchor_ranging_rounds_rsp**) – Response containing indexes of ranging rounds which failed to be configured.

2.3.200.1 Return

Error code if the command cannot be executed.

The return code will be QERR_SUCCESS if the command is valid, then the result of the command execution will be in the status field of the corresponding response.

2.3.201 fira_helper_dt_tag_configure_ranging_rounds

enum qerr **fira_helper_dt_tag_configure_ranging_rounds**(struct fira_context *ctx, uint32_t session_handle, struct *dt_tag_ranging_rounds_config* *ranging_rounds_config, struct *dt_tag_round_indexes_rsp* *round_indexes_rsp)

[Not supported in QM33 SDK] Configure ranging rounds for DT-Tag.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **ranging_rounds_config** (struct *dt_tag_ranging_rounds_config**) – Configuration parameters of the ranging rounds.
- **round_indexes_rsp** (struct *dt_tag_round_indexes_rsp**) – Indexes of ranging rounds which failed to be configured.

2.3.201.1 Return

QERR_SUCCESS or error if the command cannot be executed.

The return code will be QERR_SUCCESS if the command is valid, then the result of the command execution will be in the status field of the corresponding response.

2.3.202 fira_helper_set_session_ut_tx_interval_ms

enum qerr **fira_helper_set_session_ut_tx_interval_ms**(struct fira_context *ctx, uint32_t session_handle, uint32_t interval)

[Not supported in QM33 SDK] Set time interval between UTMs.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **interval** (uint32_t) – Time interval between UTMs (in ms).

2.3.202.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.203 fira_helper_set_session_ut_random_window

enum qerr **fira_helper_set_session_ut_random_window**(struct fira_context *ctx, uint32_t session_handle, uint32_t window)

[Not supported in QM33 SDK] Set duration of random window in which UTMs can be send.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **window** (uint32_t) – Random window for UTMs (in ms).

2.3.203.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.204 fira_helper_set_session_ut_tx_timestamp_len

enum qerr **fira_helper_set_session_ut_tx_timestamp_len**(struct fira_context *ctx, uint32_t session_handle, uint8_t len)

[Not supported in QM33 SDK] Set length of timestamp in UTMs.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **len** (uint8_t) – Length of timestamps included in UTMs not present (0, default), 40-bit timestamp (1), 64-bit timestamp (2).

2.3.204.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.205 fira_helper_set_session_ut_device_id_len

enum qerr **fira_helper_set_session_ut_device_id_len**(struct fira_context *ctx, uint32_t session_handle, uint8_t len)

[Not supported in QM33 SDK] Set UL-TDoA device id length.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **len** (uint8_t) – Length of device id not present (0, default), 16-bit (1), 32-bit timestamp (2), 64-bit timestamp (3).

2.3.205.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.206 fira_helper_set_session_ut_device_id

enum qerr **fira_helper_set_session_ut_device_id**(struct fira_context *ctx, uint32_t session_handle, uint64_t id)

[Not supported in QM33 SDK] Set value of UL-TDoA device id.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **id** (uint64_t) – Device id

2.3.206.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.207 fira_helper_set_session_ut_report_config_interval

enum qerr **fira_helper_set_session_ut_report_config_interval**(struct fira_context *ctx, uint32_t session_handle, uint8_t interval)

[Not supported in QM33 SDK] Set value of UT-Anchor report config time interval.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **interval** (uint8_t) – Time interval between reports.

2.3.207.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.208 fira_helper_set_session_ut_report_config_count

enum qerr **fira_helper_set_session_ut_report_config_count**(struct fira_context *ctx, uint32_t session_handle, uint8_t count)

[Not supported in QM33 SDK] Set value of UT-Anchor report config count.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **count** (uint8_t) – Measurement count between reports.

2.3.208.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.209 fira_helper_set_session_ut_report_config_event

enum qerr **fira_helper_set_session_ut_report_config_event**(struct fira_context *ctx, uint32_t session_handle, uint8_t event)

[Not supported in QM33 SDK] Set value of UT-Anchor report config events type.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **event** (uint8_t) – Rx event to be reported.

2.3.209.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.210 fira_helper_get_session_ut_tx_interval_ms

enum qerr **fira_helper_get_session_ut_tx_interval_ms**(struct fira_context *ctx, uint32_t session_handle, uint32_t *interval)

[Not supported in QM33 SDK] Get time interval between UTMs.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **interval** (uint32_t*) – Time interval between UTMs (in ms).

2.3.210.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.211 fira_helper_get_session_ut_random_window

enum qerr **fira_helper_get_session_ut_random_window**(struct fira_context *ctx, uint32_t session_handle, uint32_t *window)

[Not supported in QM33 SDK] Set duration of random window in which UTMs can be send.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **window** (uint32_t*) – Random window for UTMs (in ms).

2.3.211.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.212 fira_helper_get_session_ut_tx_timestamp_len

enum qerr **fira_helper_get_session_ut_tx_timestamp_len**(struct fira_context *ctx, uint32_t session_handle, uint8_t *len)

[Not supported in QM33 SDK] Get length of timestamp in UTMs.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **len** (uint8_t*) – Length of timestamps included in UTMs not present (0, default), 40-bit timestamp (1), 64-bit timestamp (2).

2.3.212.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.213 fira_helper_get_session_ut_device_id_len

enum qerr **fira_helper_get_session_ut_device_id_len**(struct fira_context *ctx, uint32_t session_handle, uint8_t *len)

[Not supported in QM33 SDK] Get UL-TDoA device id length.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **len** (uint8_t*) – Length of device id not present (0, default), 16-bit (1), 32-bit timestamp (2), 64-bit timestamp (3).

2.3.213.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.214 fira_helper_get_session_ut_device_id

enum qerr **fira_helper_get_session_ut_device_id**(struct fira_context *ctx, uint32_t session_handle, uint64_t *id)

[Not supported in QM33 SDK] Get value of UL-TDoA device id.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **id** (uint64_t*) – Device id

2.3.214.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.215 fira_helper_get_session_ut_report_config_interval

enum qerr **fira_helper_get_session_ut_report_config_interval**(struct fira_context *ctx, uint32_t session_handle, uint8_t *interval)

[Not supported in QM33 SDK] Get value of UT-Anchor report config time interval.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **interval** (uint8_t*) – Time interval between reports.

2.3.215.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.216 fira_helper_get_session_ut_report_config_count

enum qerr **fira_helper_get_session_ut_report_config_count**(struct fira_context *ctx, uint32_t session_handle, uint8_t *count)

[Not supported in QM33 SDK] Get value of UT-Anchor report config count.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **count** (uint8_t*) – Measurement count between reports.

2.3.216.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.217 fira_helper_get_session_ut_report_config_event

enum qerr **fira_helper_get_session_ut_report_config_event**(struct fira_context *ctx, uint32_t session_handle, uint8_t *event)

[Not supported in QM33 SDK] Get value of UT-Anchor report config events type.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **event** (uint8_t*) – Rx event to be reported.

2.3.217.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.218 fira_helper_session_get_data_size_in_ranging

enum qerr **fira_helper_session_get_data_size_in_ranging**(struct fira_context *ctx, uint32_t session_handle, uint16_t *size_in_ranging)

[Not supported in QM33 SDK] Get maximum data size in ranging round.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle.
- **size_in_ranging** (uint16_t*) – Data size in ranging.

2.3.218.1 Return

QERR_SUCCESS on success, an error otherwise.

2.3.219 fira_helper_set_hus_controller_config

enum qerr **fira_helper_set_hus_controller_config**(struct fira_context *ctx, uint32_t session_handle, struct *fira_hus_controller_config_cmd* *cmd)

[Not supported in QM33 SDK] Configure phases of a HUS ranging round.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle of the targeted HUS controller.
- **cmd** (struct *fira_hus_controller_config_cmd**) – Command containing configuration parameters of each phase to bind.

2.3.219.1 Return

QERR_SUCCESS or an error otherwise. The corresponding FiRa status code is present if return value > 0.

2.3.220 fira_helper_set_hus_controlee_config

enum qerr **fira_helper_set_hus_controlee_config**(struct fira_context *ctx, uint32_t session_handle, struct *fira_hus_controlee_config_cmd* *cmd)

[Not supported in QM33 SDK] Configure phases of a HUS ranging round.

Parameters

- **ctx** (struct fira_context*) – Fira helper context.
- **session_handle** (uint32_t) – Session handle of the targeted HUS controller.
- **cmd** (struct *fira_hus_controlee_config_cmd**) – Command containing configuration parameters of each phase to bind.

2.3.220.1 Return

QERR_SUCCESS or an error otherwise. The corresponding FiRa status code is present if return value > 0.

2.3.221 fira_helper_set_session_application_data_endpoint

enum qerr **fira_helper_set_session_application_data_endpoint**(struct fira_context *context, uint32_t session_handle, uint8_t application_data_endpoint)

[Not supported in QM33 SDK] Application data endpoint setter.

Parameters

- **context** (struct fira_context*) – Fira context.
- **session_handle** (uint32_t) – Session handle.

- **application_data_endpoint** (uint8_t) – Endpoint for non-secure/secure data message exchange.

2.3.221.1 Return

QERR_SUCCESS or error.

2.3.222 fira_helper_get_session_application_data_endpoint

enum qerr **fira_helper_get_session_application_data_endpoint**(struct fira_context *context, uint32_t session_handle, uint8_t *application_data_endpoint)

[Not supported in QM33 SDK] Application data endpoint getter.

Parameters

- **context** (struct fira_context*) – Fira context.
- **session_handle** (uint32_t) – Session handle.
- **application_data_endpoint** (uint8_t*) – Endpoint for non-secure/secure data message exchange.

2.3.222.1 Return

QERR_SUCCESS or error.

2.3.223 enum fira_dt_anchor_acting_role

enum **fira_dt_anchor_acting_role**

[Not supported in QM33 SDK] Internal role played by a DT-Anchor during the particular ranging round.

2.3.223.1 Definition

```
enum fira_dt_anchor_acting_role {
    FIRA_DT_ANCHOR_ACTING_RESPONDER,
    FIRA_DT_ANCHOR_ACTING_INITIATOR
};
```

2.3.223.2 Constants

FIRA_DT_ANCHOR_ACTING_RESPONDER
The DT-Anchor acts as a responder.

FIRA_DT_ANCHOR_ACTING_INITIATOR
The DT-Anchor acts as an initiator.

2.3.224 enum fira_ranging_round_usage

enum **fira_ranging_round_usage**

Ranging mode.

2.3.224.1 Definition

```
enum fira_ranging_round_usage {
    FIRA_RANGING_ROUND_USAGE_OWR_UL_TDOA,
    FIRA_RANGING_ROUND_USAGE_SSTWR_DEFERRED,
    FIRA_RANGING_ROUND_USAGE_DSTWR_DEFERRED,
    FIRA_RANGING_ROUND_USAGE_SSTWR_NON_DEFERRED,
    FIRA_RANGING_ROUND_USAGE_DSTWR_NON_DEFERRED,
    FIRA_RANGING_ROUND_USAGE_OWR_DL_TDOA,
    FIRA_RANGING_ROUND_USAGE_OWR_AOA,
    FIRA_RANGING_ROUND_USAGE_ESS_TWR_NON_DEFERRED_CONTENTION_BASED,
    FIRA_RANGING_ROUND_USAGE_ADS_TWR_CONTENTION_BASED,
    FIRA_RANGING_ROUND_USAGE_ASSIGNED,
    FIRA_RANGING_ROUND_USAGE_HYBRID_RANGING,
    FIRA_RANGING_ROUND_USAGE_MAX,
    FIRA_RANGING_ROUND_USAGE_UNDEFINED
};
```

2.3.224.2 Constants

FIRA_RANGING_ROUND_USAGE_OWR_UL_TDOA
[Not supported in QM33 SDK] One Way Ranging UL-TDoA.

FIRA_RANGING_ROUND_USAGE_SSTWR_DEFERRED
Single-Sided Two Way Ranging with Deferred Mode.

FIRA_RANGING_ROUND_USAGE_DSTWR_DEFERRED
Dual-Sided Two Way Ranging with Deferred Mode.

FIRA_RANGING_ROUND_USAGE_SSTWR_NON_DEFERRED
Single-Sided Two Way Ranging with Non-Deferred Mode.

FIRA_RANGING_ROUND_USAGE_DSTWR_NON_DEFERRED
Dual-Sided Two Way Ranging with Non-Deferred Mode.

FIRA_RANGING_ROUND_USAGE_OWR_DL_TDOA
[Not supported in QM33 SDK] One Way Ranging for DownLink Time Difference of Arrival measurement.

FIRA_RANGING_ROUND_USAGE_OWR_AOA
[Not supported in QM33 SDK] One Way Ranging for Angle of Arrival measurement

FIRA_RANGING_ROUND_USAGE_ESS_TWR_NON_DEFERRED_CONTENTION_BASED
[Not supported in QM33 SDK] Enhanced SingleSided Two Way Ranging with Non-deferred Mode for Contention-based ranging.

FIRA_RANGING_ROUND_USAGE_ADS_TWR_CONTENTION_BASED
[Not supported in QM33 SDK] Alternative Dual-Sided Two Way Ranging for Contention-based ranging.

FIRA_RANGING_ROUND_USAGE_ASSIGNED
RFU.

FIRA_RANGING_ROUND_USAGE_HYBRID_RANGING
[Not supported in QM33 SDK] Hybrid Ranging Mode.

FIRA_RANGING_ROUND_USAGE_MAX
Max value for the range check.

FIRA_RANGING_ROUND_USAGE_UNDEFINED
Ranging round usage is undefined.

2.3.225 enum fira_dl_tdoa_ranging_method

enum **fira_dl_tdoa_ranging_method**
[Not supported in QM33 SDK] Ranging method used by DT-Anchors in DL-TDoA.

2.3.225.1 Definition

```
enum fira_dl_tdoa_ranging_method {
    FIRA_DL_TDOA_RANGING_METHOD_SSTWR,
    FIRA_DL_TDOA_RANGING_METHOD_DSTWR
};
```

2.3.225.2 Constants

FIRA_DL_TDOA_RANGING_METHOD_SSTWR
Single-Sided Two Way Ranging.

FIRA_DL_TDOA_RANGING_METHOD_DSTWR
Dual-Sided Two Way Ranging.

2.3.226 enum fira_multi_node_mode

enum **fira_multi_node_mode**
Multi-node mode.

2.3.226.1 Definition

```
enum fira_multi_node_mode {
    FIRA_MULTI_NODE_MODE_UNICAST,
    FIRA_MULTI_NODE_MODE_ONE_TO_MANY,
    FIRA_MULTI_NODE_MODE_MANY_TO_MANY,
    FIRA_MULTI_NODE_MODE_UNDEFINED
};
```

2.3.226.2 Constants

FIRA_MULTI_NODE_MODE_UNICAST
Ranging between one initiator and one responder.

FIRA_MULTI_NODE_MODE_ONE_TO_MANY
Ranging between one initiator and multiple responders.

FIRA_MULTI_NODE_MODE_MANY_TO_MANY
[Not supported in QM33 SDK] Ranging between multiple initiators and multiple responders.

FIRA_MULTI_NODE_MODE_UNDEFINED
Ranging mode is undefined.

2.3.227 enum fira_measurement_report_originator

enum **fira_measurement_report_originator**
Originator (author) of the Ranging Measurement Report Message (MRM).

2.3.227.1 Definition

```
enum fira_measurement_report_originator {
    FIRA_MEASUREMENT_REPORT_FROM_INITIATOR,
    FIRA_MEASUREMENT_REPORT_FROM_RESPONDER
};
```

2.3.227.2 Constants

FIRA_MEASUREMENT_REPORT_FROM_INITIATOR
The initiator sends a measurement report message.

FIRA_MEASUREMENT_REPORT_FROM_RESPONDER
The responder sends a measurement report message.

2.3.228 enum fira_measurement_report_type

enum **fira_measurement_report_type**
Type of the current Ranging Measurement Report Message (MRM).

2.3.228.1 Definition

```
enum fira_measurement_report_type {
    FIRA_MEASUREMENT_REPORT_TYPE_1,
    FIRA_MEASUREMENT_REPORT_TYPE_2,
    FIRA_MEASUREMENT_REPORT_TYPE_3
};
```

2.3.228.2 Constants

FIRA_MEASUREMENT_REPORT_TYPE_1
Measurement report message type 1.

FIRA_MEASUREMENT_REPORT_TYPE_2
Measurement report message type 2.

FIRA_MEASUREMENT_REPORT_TYPE_3
[Not supported in QM33 SDK] Measurement report message type 3.

2.3.229 enum fira_owr_message_type

enum **fira_owr_message_type**

[Not supported in QM33 SDK] Type of the current One Way Ranging (OWR) Message (FiRa v2.0 5.9.13 OWR Message).

2.3.229.1 Definition

```
enum fira_owr_message_type {
    FIRA_OWR_MESSAGE_TYPE_UT_BLINK,
    FIRA_OWR_MESSAGE_TYPE_UT_SYNC,
    FIRA_OWR_MESSAGE_TYPE_DT_POLL,
    FIRA_OWR_MESSAGE_TYPE_DT_RESPONSE,
    FIRA_OWR_MESSAGE_TYPE_DT_FINAL,
    FIRA_OWR_MESSAGE_TYPE_AOA_MEASUREMENT
};
```

2.3.229.2 Constants

FIRA_OWR_MESSAGE_TYPE_UT_BLINK
UL-TDOA Blink Message.

FIRA_OWR_MESSAGE_TYPE_UT_SYNC
UL-TDOA Synchronization Message.

FIRA_OWR_MESSAGE_TYPE_DT_POLL
DL-TDOA Poll Message.

FIRA_OWR_MESSAGE_TYPE_DT_RESPONSE
DL-TDOA Response Message.

FIRA_OWR_MESSAGE_TYPE_DT_FINAL
DL-TDOA Final Message.

FIRA_OWR_MESSAGE_TYPE_AOA_MEASUREMENT
AoA Measurement Message.

2.3.230 enum fira_schedule_mode

enum **fira_schedule_mode**

Slot scheduling mode used during the ranging session.

2.3.230.1 Definition

```
enum fira_schedule_mode {
    FIRA_SCHEDULE_MODE_CONTENTION_BASED,
    FIRA_SCHEDULE_MODE_TIME_SCHEDULED,
    FIRA_SCHEDULE_MODE_HYBRID_BASED,
    FIRA_SCHEDULE_MODE_MAX,
    FIRA_SCHEDULE_MODE_UNDEFINED
};
```

2.3.230.2 Constants

FIRA_SCHEDULE_MODE_CONTENTION_BASED
[Not supported in QM33 SDK] Contention-based ranging.

FIRA_SCHEDULE_MODE_TIME_SCHEDULED
Time-scheduled ranging.

FIRA_SCHEDULE_MODE_HYBRID_BASED
[Not supported in QM33 SDK] Hybrid-based ranging.

FIRA_SCHEDULE_MODE_MAX
Max value defined

FIRA_SCHEDULE_MODE_UNDEFINED
Scheduled mode is undefined.

2.3.231 enum fira_rframe_config

enum **fira_rframe_config**

Rframe configuration used to transmit/receive ranging messages.

2.3.231.1 Definition

```
enum fira_rframe_config {
    FIRA_RFRAME_CONFIG_SP0,
    FIRA_RFRAME_CONFIG_SP1,
    FIRA_RFRAME_CONFIG_SP2,
    FIRA_RFRAME_CONFIG_SP3
};
```

2.3.231.2 Constants

FIRA_RFRAME_CONFIG_SP0
Use SP0 mode. (Applicable only for PCTT)

FIRA_RFRAME_CONFIG_SP1
Use SP1 mode.

FIRA_RFRAME_CONFIG_SP2
RFU

FIRA_RFRAME_CONFIG_SP3
Use SP3 mode.

2.3.232 enum fira_prf_mode

enum **fira_prf_mode**
Pulse Repetition Frequency mode

2.3.232.1 Definition

```
enum fira_prf_mode {
    FIRA_PRF_MODE_BPRF,
    FIRA_PRF_MODE_HPRF,
    FIRA_PRF_MODE_HPRF_HIGH_RATE
};
```

2.3.232.2 Constants

FIRA_PRF_MODE_BPRF
Base Pulse Repetition Frequency.

FIRA_PRF_MODE_HPRF
[Not supported in QM33 SDK] Higher Pulse Repetition Frequency.

FIRA_PRF_MODE_HPRF_HIGH_RATE
[Not supported in QM33 SDK] Higher Pulse Repetition Frequency allows high data rate (27.2 Mbps and 31.2 Mbps).

2.3.232.3 Description

This enum is not used in the current implementation.

2.3.233 enum fira_preamble_duration

enum **fira_preamble_duration**

Duration of preamble in symbols.

2.3.233.1 Definition

```
enum fira_preamble_duration {
    FIRA_PREAMBLE_DURATION_32,
    FIRA_PREAMBLE_DURATION_64
};
```

2.3.233.2 Constants

FIRA_PREAMBLE_DURATION_32
[Not supported in QM33 SDK] 32 symbols duration.

FIRA_PREAMBLE_DURATION_64
64 symbols duration.

2.3.234 enum fira_sfd_id

enum **fira_sfd_id**

Start-of-frame delimiter.

2.3.234.1 Definition

```
enum fira_sfd_id {
    FIRA_SFD_ID_0,
    FIRA_SFD_ID_1,
    FIRA_SFD_ID_2,
    FIRA_SFD_ID_3,
    FIRA_SFD_ID_4
};
```

2.3.234.2 Constants

FIRA_SFD_ID_0
Delimiter is [0 +1 0 –1 +1 0 0 –1]

FIRA_SFD_ID_1
[Not supported in QM33 SDK] Delimiter is [–1 –1 +1 –1]

FIRA_SFD_ID_2
Delimiter is [–1 –1 –1 +1 –1 –1 +1 –1]

FIRA_SFD_ID_3
[Not supported in QM33 SDK] Delimiter is [–1 –1 –1 –1 –1 +1 +1 –1 –1 +1 –1 +1 –1 –1 +1 –1]

FIRA_SFD_ID_4
[Not supported in QM33 SDK] Delimiter is [–1 –1 –1 –1 –1 –1 –1 +1 –1 –1 +1 –1 –1 +1 –1 +1 –1 +1 –1 –1 –1 +1 +1 –1 –1 –1 +1 –1 +1 +1 –1 –1]

2.3.235 enum fira_sts_segments

enum **fira_sts_segments**

Number of STS segments.

2.3.235.1 Definition

```
enum fira_sts_segments {
    FIRA_STS_SEGMENTS_0,
    FIRA_STS_SEGMENTS_1,
    FIRA_STS_SEGMENTS_2,
    FIRA_STS_SEGMENTS_3,
    FIRA_STS_SEGMENTS_4
};
```

2.3.235.2 Constants

FIRA_STS_SEGMENTS_0
No STS Segment (Rframe config SP0).

FIRA_STS_SEGMENTS_1
1 STS Segment.

FIRA_STS_SEGMENTS_2
[Not supported in QM33 SDK] 2 STS Segments.

FIRA_STS_SEGMENTS_3
[Not supported in QM33 SDK] 3 STS Segments.

FIRA_STS_SEGMENTS_4
[Not supported in QM33 SDK] 4 STS Segments.

2.3.236 enum fira_psdu_data_rate

enum **fira_psdu_data_rate**

Data rate used to exchange PSDUs.

2.3.236.1 Definition

```
enum fira_psdu_data_rate {
    FIRA_PSDU_DATA_RATE_6M81,
    FIRA_PSDU_DATA_RATE_7M80,
    FIRA_PSDU_DATA_RATE_27M2,
    FIRA_PSDU_DATA_RATE_31M2
};
```

2.3.236.2 Constants

FIRA_PSDU_DATA_RATE_6M81
: 6.8Mb/s rate.

FIRA_PSDU_DATA_RATE_7M80
: *[Not supported in QM33 SDK]* 7.8Mb/s rate.

FIRA_PSDU_DATA_RATE_27M2
: *[Not supported in QM33 SDK]* 27.2Mb/s rate.

FIRA_PSDU_DATA_RATE_31M2
: *[Not supported in QM33 SDK]* 31.2Mb/s rate.

2.3.237 enum fira_phr_data_rate

enum **fira_phr_data_rate**
: Data rate used to exchange PHR.

2.3.237.1 Definition

```
enum fira_phr_data_rate {
    FIRA_PHR_DATA_RATE_850K,
    FIRA_PHR_DATA_RATE_6M81
};
```

2.3.237.2 Constants

FIRA_PHR_DATA_RATE_850K
: 850kb/s rate.

FIRA_PHR_DATA_RATE_6M81
: 6.8Mb/s rate.

2.3.238 enum fira_mac_fcs_type

enum **fira_mac_fcs_type**
: Length of Frame Check Sequence.

2.3.238.1 Definition

```
enum fira_mac_fcs_type {
    FIRA_MAC_FCS_TYPE_CRC_16,
    FIRA_MAC_FCS_TYPE_CRC_32
};
```

2.3.238.2 Constants

FIRA_MAC_FCS_TYPE_CRC_16
2 bytes sequence.

FIRA_MAC_FCS_TYPE_CRC_32
4 bytes sequence.

2.3.239 enum fira_data_transfer_status

enum **fira_data_transfer_status**
[Not supported in QM33 SDK] Data transfer status.

2.3.239.1 Definition

```
enum fira_data_transfer_status {
    FIRA_DATA_TRANSFER_STATUS_REPETITION_OK,
    FIRA_DATA_TRANSFER_STATUS_OK,
    FIRA_DATA_TRANSFER_STATUS_ERROR_DATA_TRANSFER,
    FIRA_DATA_TRANSFER_STATUS_ERROR_NO_CREDIT_AVAILABLE,
    FIRA_DATA_TRANSFER_STATUS_ERROR_REJECTED,
    FIRA_DATA_TRANSFER_STATUS_SESSION_TYPE_NOT_SUPPORTED,
    FIRA_DATA_TRANSFER_STATUS_ERROR_DATA_TRANSFER_IS_ONGOING,
    FIRA_DATA_TRANSFER_STATUS_INVALID_FORMAT
};
```

2.3.239.2 Constants

FIRA_DATA_TRANSFER_STATUS_REPETITION_OK
If DATA_REPETITION_COUNT>0 and if SESSION_DATA_TRANSFER_STATUS_NTF_CONFIG = Enable; it indicates that one Data transmission is completed in a RR.

FIRA_DATA_TRANSFER_STATUS_OK
For TWR - it indicates that the Application Data transmission is completed. For OWR - it indicates that the Application Data transmission and its repetitions of DATA_REPETITION_COUNT is completed.

FIRA_DATA_TRANSFER_STATUS_ERROR_DATA_TRANSFER
Application Data couldn't be sent due to an unrecoverable error.

FIRA_DATA_TRANSFER_STATUS_ERROR_NO_CREDIT_AVAILABLE
DATA_MESSAGE_SND is not accepted as no credit is available.

FIRA_DATA_TRANSFER_STATUS_ERROR_REJECTED
DATA_MESSAGE_SND packet sent in wrong state or Application Data Size exceeds the maximum size that can be sent in one Ranging Round.

FIRA_DATA_TRANSFER_STATUS_SESSION_TYPE_NOT_SUPPORTED
Data transfer is not supported for given session type.

FIRA_DATA_TRANSFER_STATUS_ERROR_DATA_TRANSFER_IS_ONGOING
Application Data is being transmitted and the number of configured DATA_REPETITION_COUNT transmissions is not yet completed.

FIRA_DATA_TRANSFER_STATUS_INVALID_FORMAT
The format of the command DATA_MESSAGE_SND associated with this notification is incorrect (e.g, a parameter is missing, a parameter value is invalid).

2.3.240 enum fira_measurement_type

enum **fira_measurement_type**

The different type of available measurements.

2.3.240.1 Definition

```
enum fira_measurement_type {
    FIRA_MEASUREMENT_TYPE_RANGE,
    FIRA_MEASUREMENT_TYPE_AOA,
    FIRA_MEASUREMENT_TYPE_AOA_AZIMUTH,
    FIRA_MEASUREMENT_TYPE_AOA_ELEVATION,
    FIRA_MEASUREMENT_TYPE_AOA_AZIMUTH_ELEVATION,
    __FIRA_MEASUREMENT_TYPE_AFTER_LAST
};
```

2.3.240.2 Constants

FIRA_MEASUREMENT_TYPE_RANGE
Measure only range.

FIRA_MEASUREMENT_TYPE_AOA
Measure range + unspecified AoA.

FIRA_MEASUREMENT_TYPE_AOA_AZIMUTH
Measure range + azimuth.

FIRA_MEASUREMENT_TYPE_AOA_ELEVATION
Measure range + elevation.

FIRA_MEASUREMENT_TYPE_AOA_AZIMUTH_ELEVATION
Measure range+azimuth+elevation.

__FIRA_MEASUREMENT_TYPE_AFTER_LAST
Internal use.

2.3.241 struct fira_session_time_base

struct **fira_session_time_base**

[Not supported in QM33 SDK] Session time base information.

2.3.241.1 Definition

```
struct fira_session_time_base {
    uint8_t config;
    uint32_t session_handle;
    uint32_t time_offset_us;
}
```

2.3.241.2 Members

config
Time base configuration bitfield. b0: 1:enable - 0:disable b1: 1:continue - 0:stop b2: 1:resync - 0:no resync

session_handle
Session handle of the reference session.

time_offset_us
Time offset in microseconds.

2.3.242 struct fira_measurement_sequence_step

struct **fira_measurement_sequence_step**
One measurement step configuration.

2.3.242.1 Definition

```
struct fira_measurement_sequence_step {
    enum fira_measurement_type type;
    uint8_t n_measurements;
    int8_t rx_ant_set_nonranging;
    int8_t rx_ant_sets_ranging[2];
    int8_t tx_ant_set_nonranging;
    int8_t tx_ant_set_ranging;
}
```

2.3.242.2 Members

type
The type of measurement.

n_measurements
The number of measurement to do.

rx_ant_set_nonranging
RX antenna set for non-ranging frames.

rx_ant_sets_ranging
RX antenna sets for ranging frames.

tx_ant_set_nonranging
TX antenna set for non-ranging frames.

tx_ant_set_ranging
TX antenna set for ranging frames.

2.3.243 struct fira_measurement_sequence

struct **fira_measurement_sequence**

Measurement sequence configuration.

2.3.243.1 Definition

```
struct fira_measurement_sequence {
    uint8_t n_steps;
    struct fira_measurement_sequence_step steps[];
}
```

2.3.243.2 Members

n_steps
Number of steps in the sequence.

steps
Array of step configuration.

2.3.244 enum fira_ranging_diagnostics_frame_report_flags

enum **fira_ranging_diagnostics_frame_report_flags**

Activation flags for different frame diagnostics information.

2.3.244.1 Definition

```
enum fira_ranging_diagnostics_frame_report_flags {
    FIRA_RANGING_DIAGNOSTICS_FRAME_REPORT_NONE,
    FIRA_RANGING_DIAGNOSTICS_FRAME_REPORT_AOAS,
    FIRA_RANGING_DIAGNOSTICS_FRAME_REPORT_CFO,
    FIRA_RANGING_DIAGNOSTICS_FRAME_REPORT_EMITTER_SHORT_ADDR,
    FIRA_RANGING_DIAGNOSTICS_FRAME_REPORT_SEGMENT_METRICS,
    FIRA_RANGING_DIAGNOSTICS_FRAME_REPORT_CIRS,
    __FIRA_RANGING_DIAGNOSTICS_FRAME_REPORT_AFTER_LAST
};
```

2.3.244.2 Constants

FIRA_RANGING_DIAGNOSTICS_FRAME_REPORT_NONE
No specific frame diagnostic report requested.

FIRA_RANGING_DIAGNOSTICS_FRAME_REPORT_AOAS
Report AOA in frame diagnostics.

FIRA_RANGING_DIAGNOSTICS_FRAME_REPORT_CFO
Report Clock Frequency Offset in report, as measured on the first Rx RFRAME in the round.

FIRA_RANGING_DIAGNOSTICS_FRAME_REPORT_EMITTER_SHORT_ADDR
Report the MAC emitter short address in frame diagnostics.

FIRA_RANGING_DIAGNOSTICS_FRAME_REPORT_SEGMENT_METRICS
Report Segment Metrics in frame diagnostics.

FIRA_RANGING_DIAGNOSTICS_FRAME_REPORT_CIRS
Report CIR in frame diagnostics.

__FIRA_RANGING_DIAGNOSTICS_FRAME_REPORT_AFTER_LAST
Internal use.

2.3.245 enum fira_ranging_diagnostics_frame_reports_status_flags

enum **fira_ranging_diagnostics_frame_reports_status_flags**
Flags for the invidual frame report's status bitfield.

2.3.245.1 Definition

```
enum fira_ranging_diagnostics_frame_reports_status_flags {
    FIRA_RANGING_DIAGNOSTICS_FRAME_REPORTS_STATUS_FLAGS_SUCCESS,
    FIRA_RANGING_DIAGNOSTICS_FRAME_REPORTS_STATUS_FLAGS_WIFI_COEX,
    FIRA_RANGING_DIAGNOSTICS_FRAME_REPORTS_STATUS_FLAGS_GRANT_DURATION_EXCEEDED
};
```

2.3.245.2 Constants

FIRA_RANGING_DIAGNOSTICS_FRAME_REPORTS_STATUS_FLAGS_SUCCESS
False when then frame Rx has failed for some reason. Always true for Tx.

FIRA_RANGING_DIAGNOSTICS_FRAME_REPORTS_STATUS_FLAGS_WIFI_COEX
True if the Wifi Coex GPIO was on while transmitting the frame.

FIRA_RANGING_DIAGNOSTICS_FRAME_REPORTS_STATUS_FLAGS_GRANT_DURATION_EXCEEDED
True if the MAX_GRANT_DURATION has been exceeded.

2.3.246 enum fira_sts_length

enum **fira_sts_length**
Number of symbols in a STS segment.

2.3.246.1 Definition

```
enum fira_sts_length {
    FIRA_STS_LENGTH_32,
    FIRA_STS_LENGTH_64,
    FIRA_STS_LENGTH_128
};
```

2.3.246.2 Constants

FIRA_STS_LENGTH_32
The STS length is 32 symbols.

FIRA_STS_LENGTH_64
The STS length is 64 symbols.

FIRA_STS_LENGTH_128
The STS length is 128 symbols.

2.3.247 enum fira_session_info_ntf_config

enum **fira_session_info_ntf_config**
[Not supported in QM33 SDK] Configure session info notification.

2.3.247.1 Definition

```
enum fira_session_info_ntf_config {
    FIRA_SESSION_INFO_NTF_CONFIG_DISABLED,
    FIRA_SESSION_INFO_NTF_CONFIG_ALWAYS,
    FIRA_SESSION_INFO_NTF_CONFIG_PROXIMITY,
    FIRA_SESSION_INFO_NTF_CONFIG_AOA,
    FIRA_SESSION_INFO_NTF_CONFIG_PROXIMITY_AND_AOA,
    FIRA_SESSION_INFO_NTF_CONFIG_PROXIMITY_CROSSING,
    FIRA_SESSION_INFO_NTF_CONFIG_AOA_CROSSING,
    FIRA_SESSION_INFO_NTF_CONFIG_PROXIMITY_AND_AOA_CROSSING
};
```

2.3.247.2 Constants

FIRA_SESSION_INFO_NTF_CONFIG_DISABLED
Do not report range data.

FIRA_SESSION_INFO_NTF_CONFIG_ALWAYS
Report range data.

FIRA_SESSION_INFO_NTF_CONFIG_PROXIMITY
Report range data if it is within proximity range defined by proximity parameters (NEAR_PROXIMITY_CONFIG/FAR).

FIRA_SESSION_INFO_NTF_CONFIG_AOA
Report range data in AoA upper and lower bound. defined by AOA parameters (FIRA_SESSION_PARAM_ATTR_SESSION_INFO_NTF_UPPER/LOWER_BOUND_AOA_AZIMUTH/ELEVATION)

FIRA_SESSION_INFO_NTF_CONFIG_PROXIMITY_AND_AOA
Report range data in AoA upper and lower bound as well as in proximity range.

FIRA_SESSION_INFO_NTF_CONFIG_PROXIMITY_CROSSING
Same as FIRA_SESSION_INFO_NTF_CONFIG_PROXIMITY, but issues notification on crossing of boundaries. As for now, same notif is sent for “enter” and “exit” events.

FIRA_SESSION_INFO_NTF_CONFIG_AOA_CROSSING
Same as FIRA_SESSION_INFO_NTF_CONFIG_AOA, but issues notification on crossing of boundaries. As for now, same notif is sent for “enter” and “exit” events.

FIRA_SESSION_INFO_NTF_CONFIG_PROXIMITY_AND_AOA_CROSSING
Same as FIRA_SESSION_INFO_NTF_CONFIG_PROXIMITY_AND_AOA, but issues notification on crossing of boundaries. As for now, same notif is sent for "enter" and "exit" events.

2.3.248 enum fira_link_layer_mode

enum **fira_link_layer_mode**

Link layer behavior.

2.3.248.1 Definition

```
enum fira_link_layer_mode {
    FIRA_LINK_LAYER_MODE_BYPASS,
    FIRA_LINK_LAYER_MODE_CONNECTION_LESS
};
```

2.3.248.2 Constants

FIRA_LINK_LAYER_MODE_BYPASS
No link layer overhead is added to the MDSDU.

FIRA_LINK_LAYER_MODE_CONNECTION_LESS
[Not supported in QM33 SDK] Link layer header is needed to provide addressing capabilities for data transmission.

2.3.249 enum fira_message_id

enum **fira_message_id**

Message identifiers, used in report and in messages.

2.3.249.1 Definition

```
enum fira_message_id {
    FIRA_MESSAGE_ID_RANGING_INITIATION,
    FIRA_MESSAGE_ID_RANGING_RESPONSE,
    FIRA_MESSAGE_ID_RANGING_FINAL,
    FIRA_MESSAGE_ID_CONTROL,
    FIRA_MESSAGE_ID_MEASUREMENT_REPORT,
    FIRA_MESSAGE_ID_RESULT_REPORT,
    FIRA_MESSAGE_ID_CONTROL_UPDATE,
    FIRA_MESSAGE_ID_ONE_WAY_RANGING,
    FIRA_MESSAGE_ID_DATA,
    FIRA_MESSAGE_ID_RFRAME_MAX,
    FIRA_MESSAGE_ID_MAX
};
```

2.3.249.2 Constants

FIRA_MESSAGE_ID_RANGING_INITIATION
Ranging Initiation Message.

FIRA_MESSAGE_ID_RANGING_RESPONSE
Ranging Response Message.

FIRA_MESSAGE_ID_RANGING_FINAL
Ranging Final Message, only for DS-TWR.

FIRA_MESSAGE_ID_CONTROL
Control Message, sent by the controller.

FIRA_MESSAGE_ID_MEASUREMENT_REPORT
Measurement Report Message.

FIRA_MESSAGE_ID_RESULT_REPORT
Ranging Result Report Message.

FIRA_MESSAGE_ID_CONTROL_UPDATE
Control Update Message.

FIRA_MESSAGE_ID_ONE_WAY_RANGING
[Not supported in QM33 SDK] One Way Ranging Message (see internal types).

FIRA_MESSAGE_ID_DATA
[Not supported in QM33 SDK] Data Message.

FIRA_MESSAGE_ID_RFRAME_MAX
Maximum identifier of messages transmitted as an RFRAME (without a payload).

FIRA_MESSAGE_ID_MAX
Maximum identifier of all messages.

2.3.250 enum fira_result_report_config_flags

enum **fira_result_report_config_flags**
result report config flags.

2.3.250.1 Definition

```
enum fira_result_report_config_flags {
    FIRA_RESULT_REPORT_CONFIG_REPORT_TOF,
    FIRA_RESULT_REPORT_CONFIG_REPORT_AOA_AZIMUTH,
    FIRA_RESULT_REPORT_CONFIG_REPORT_AOA_ELEVATION,
    FIRA_RESULT_REPORT_CONFIG_REPORT_AOA_FOM,
    FIRA_RESULT_REPORT_CONFIG_REPORT_ALL
};
```

2.3.250.2 Constants

FIRA_RESULT_REPORT_CONFIG_REPORT_TOF
Report Time of flight.

FIRA_RESULT_REPORT_CONFIG_REPORT_AOA_AZIMUTH
Report azimuth angle.

FIRA_RESULT_REPORT_CONFIG_REPORT_AOA_ELEVATION
Report elevation angle.

FIRA_RESULT_REPORT_CONFIG_REPORT_AOA_FOM
report AoA figure of merit.

FIRA_RESULT_REPORT_CONFIG_REPORT_ALL
Maximum value of the parameter.

2.3.251 enum fira_data_message_status

enum **fira_data_message_status**
[Not supported in QM33 SDK] status of data message receive notification.

2.3.251.1 Definition

```
enum fira_data_message_status {
    FIRA_DATA_MESSAGE_STATUS_SUCCESS,
    FIRA_DATA_MESSAGE_STATUS_FAILED,
    FIRA_DATA_MESSAGE_STATUS_UNKNOWN
};
```

2.3.251.2 Constants

FIRA_DATA_MESSAGE_STATUS_SUCCESS
All data segments in the round were successfully received.

FIRA_DATA_MESSAGE_STATUS_FAILED
Intended operation failed to complete, e.g. incomplete ranging round with piggyback data.

FIRA_DATA_MESSAGE_STATUS_UNKNOWN
Failure due to unknown reason.

2.3.252 enum fira_dt_location_coord_system_type

enum **fira_dt_location_coord_system_type**
[Not supported in QM33 SDK] Coordinate System Type of a DT-Anchor.

2.3.252.1 Definition

```
enum fira_dt_location_coord_system_type {
    FIRA_DT_LOCATION_COORD_WGS84,
    FIRA_DT_LOCATION_COORD_RELATIVE,
    FIRA_DT_LOCATION_COORD_INVALID
};
```

2.3.252.2 Constants

FIRA_DT_LOCATION_COORD_WGS84
The location is given in WGS84 coordinate system (longitude, latitude, altitude,(see `struct fira_wgs84_location`).

FIRA_DT_LOCATION_COORD_RELATIVE
The location is given in relative coordinates system (see `struct fira_relative_location`).

FIRA_DT_LOCATION_COORD_INVALID
is a value in RSU range for test.

2.3.253 enum fira_owr_dtm_timestamp_type

enum **`fira_owr_dtm_timestamp_type`**
[Not supported in QM33 SDK] DTM TX Timestamp type.

2.3.253.1 Definition

```
enum fira_owr_dtm_timestamp_type {
    FIRA_OWR_DTM_TIMESTAMP_LOCAL_TIME_BASE,
    FIRA_OWR_DTM_TIMESTAMP_COMMON_TIME_BASE
};
```

2.3.253.2 Constants

FIRA_OWR_DTM_TIMESTAMP_LOCAL_TIME_BASE
timestamp in local time base.

FIRA_OWR_DTM_TIMESTAMP_COMMON_TIME_BASE
timestamp in common time base of the Initiator DT-Anchor.

2.3.254 enum fira_owr_dtm_timestamp_len

enum **`fira_owr_dtm_timestamp_len`**
[Not supported in QM33 SDK] DTM TX Timestamp length.

2.3.254.1 Definition

```
enum fira_owr_dtm_timestamp_len {
    FIRA_OWR_DTM_TIMESTAMP_40BITS,
    FIRA_OWR_DTM_TIMESTAMP_64BITS
};
```

2.3.254.2 Constants

FIRA_OWR_DTM_TIMESTAMP_40BITS
40 bits timestamp.

FIRA_OWR_DTM_TIMESTAMP_64BITS
64 bits timestamp.

2.3.255 enum fira_owr_utm_timestamp_len

enum **fira_owr_utm_timestamp_len**
[Not supported in QM33 SDK] UTM TX Timestamp length.

2.3.255.1 Definition

```
enum fira_owr_utm_timestamp_len {
    FIRA_OWR_UTM_TIMESTAMP_NOT_PRESENT,
    FIRA_OWR_UTM_TIMESTAMP_40BITS,
    FIRA_OWR_UTM_TIMESTAMP_64BITS
};
```

2.3.255.2 Constants

FIRA_OWR_UTM_TIMESTAMP_NOT_PRESENT
0 bits timestamp.

FIRA_OWR_UTM_TIMESTAMP_40BITS
40 bits timestamp.

FIRA_OWR_UTM_TIMESTAMP_64BITS
64 bits timestamp.

2.3.256 enum fira_ranging_round_control_flags

enum **fira_ranging_round_control_flags**
Ranging round control flags. Below bits make sense when SCHEDULE_MODE is set to Time scheduled.

2.3.256.1 Definition

```
enum fira_ranging_round_control_flags {
    FIRA_RANGING_ROUND_CONTROL_RRRM_EXPECTED,
    FIRA_RANGING_ROUND_CONTROL_CM_EXPECTED,
    FIRA_RANGING_ROUND_CONTROL_RCP_EXCLUDED,
    FIRA_RANGING_ROUND_CONTROL_MEASUREMENT_REPORT_PHASE,
    FIRA_RANGING_ROUND_CONTROL_MEASUREMENT_REPORT_MESSAGE,
    FIRA_RANGING_ROUND_CONTROL_ALL
};
```

2.3.256.2 Constants

FIRA_RANGING_ROUND_CONTROL_RRRM_EXPECTED
If set to 1, a Controller shall schedule an RRRM in the Ranging Device Management List (RDML). If set to 0, a Controller shall not schedule an RRRM in the RDML.

FIRA_RANGING_ROUND_CONTROL_CM_EXPECTED
If set to 1, a Controller shall send a CM in-band and a Controlee shall expect a CM in-band. If set to 0, a Controller shall not send a CM in-band and a Controlee shall not expect a CM in-band.

FIRA_RANGING_ROUND_CONTROL_RCP_EXCLUDED
If set to 1, RCP is excluded in Ranging Round, means CM is piggybacked with the RIM. If set to 0, RCP is included in Ranging Round.

FIRA_RANGING_ROUND_CONTROL_MEASUREMENT_REPORT_PHASE
UWBS shall ignore this bit.

FIRA_RANGING_ROUND_CONTROL_MEASUREMENT_REPORT_MESSAGE
If set to 1, the controller shall schedule the MRM to be sent from the responder(s) to the initiator in the RDML. If set to 0, the controller shall schedule the MRM to be sent from the initiator to the Responder(s) in the RDML.

FIRA_RANGING_ROUND_CONTROL_ALL
Maximum value of the parameter.

2.3.257 enum fira_owr_utm_device_id_len

enum **fira_owr_utm_device_id_len**
[Not supported in QM33 SDK] UTM Device ID length.

2.3.257.1 Definition

```
enum fira_owr_utm_device_id_len {
    FIRA_OWR_UTM_DEVICE_ID_NOT_PRESENT,
    FIRA_OWR_UTM_DEVICE_ID_16BITS,
    FIRA_OWR_UTM_DEVICE_ID_32BITS,
    FIRA_OWR_UTM_DEVICE_ID_64BITS
};
```

2.3.257.2 Constants

FIRA_OWR_UTM_DEVICE_ID_NOT_PRESENT
0 bits device id.

FIRA_OWR_UTM_DEVICE_ID_16BITS
16 bits device id.

FIRA_OWR_UTM_DEVICE_ID_32BITS
32 bits device id.

FIRA_OWR_UTM_DEVICE_ID_64BITS
64 bits device id.

2.3.258 enum fira_multicast_update_status

enum **fira_multicast_update_status**
controlee change status after update controller multicast list command.

2.3.258.1 Definition

```
enum fira_multicast_update_status {
    FIRA_MULTICAST_UPDATE_STATUS_OK_MULTICAST_LIST_UPDATE,
    FIRA_MULTICAST_UPDATE_STATUS_ERROR_MULTICAST_LIST_FULL,
    FIRA_MULTICAST_UPDATE_STATUS_ERROR_KEY_FETCH_FAIL,
    FIRA_MULTICAST_UPDATE_STATUS_ERROR_SUB_SESSION_ID_NOT_FOUND,
    FIRA_MULTICAST_UPDATE_STATUS_ERROR_SUB_SESSION_KEY_NOT_FOUND,
    FIRA_MULTICAST_UPDATE_STATUS_ERROR_SUB_SESSION_KEY_NOT_APPLICABLE,
    FIRA_MULTICAST_UPDATE_STATUS_ERROR_SESSION_KEY_NOT_FOUND,
    FIRA_MULTICAST_UPDATE_STATUS_ERROR_ADDRESS_NOT_FOUND,
    FIRA_MULTICAST_UPDATE_STATUS_ERROR_ADDRESS_ALREADY_PRESENT
};
```

2.3.258.2 Constants

FIRA_MULTICAST_UPDATE_STATUS_OK_MULTICAST_LIST_UPDATE
it shall be reported if the multicast list is updated (Add/Delete) successfully for the given Controlee.

FIRA_MULTICAST_UPDATE_STATUS_ERROR_MULTICAST_LIST_FULL
it shall be reported for a Controlee if the multicast is full.

FIRA_MULTICAST_UPDATE_STATUS_ERROR_KEY_FETCH_FAIL
it shall be reported for a Controlee if Session Key fetch from Secure Component is failed.

FIRA_MULTICAST_UPDATE_STATUS_ERROR_SUB_SESSION_ID_NOT_FOUND
it shall be reported for a Controlee if Sub-Session ID is not found in Secure Component.

FIRA_MULTICAST_UPDATE_STATUS_ERROR_SUB_SESSION_KEY_NOT_FOUND
it shall be reported for a Controlee if Sub-Session Key is not found in Secure Component.

FIRA_MULTICAST_UPDATE_STATUS_ERROR_SUB_SESSION_KEY_NOT_APPLICABLE
it shall be reported for a Controlee if Sub-Session Key is configured with STS config is other than 0x04 (Provisioned STS for Responder specific Sub-session Key).

FIRA_MULTICAST_UPDATE_STATUS_ERROR_SESSION_KEY_NOT_FOUND
it shall be reported for a Controlee if Sub-Session Key is configured but SESSION_KEY App configuration parameter is not programmed.

FIRA_MULTICAST_UPDATE_STATUS_ERROR_ADDRESS_NOT_FOUND
it shall be reported for a Controlee if its short address is not found while deleting its entry from multicast list.

FIRA_MULTICAST_UPDATE_STATUS_ERROR_ADDRESS_ALREADY_PRESENT
it shall be reported for a Controlee if its short address is already present in the multicast list.

2.3.259 enum fira_data_segment_info

enum **fira_data_segment_info**
[Not supported in QM33 SDK] information about data packet received.

2.3.259.1 Definition

```
enum fira_data_segment_info {
    FIRA_DATA_SEGMENT_FIRST,
    FIRA_DATA_SEGMENT_LAST,
    __FIRA_DATA_SEGMENT_AFTER_LAST
};
```

2.3.259.2 Constants

FIRA_DATA_SEGMENT_FIRST
packet is the first one in data message.

FIRA_DATA_SEGMENT_LAST
packet is the last one in data message.

__FIRA_DATA_SEGMENT_AFTER_LAST
Internal use.

2.3.260 enum fira_owr_aoa_measurement_ntf_period

enum **fira_owr_aoa_measurement_ntf_period**
[Not supported in QM33 SDK] period of sending SESSION_INFO_NTF.

2.3.260.1 Definition

```
enum fira_owr_aoa_measurement_ntf_period {
    FIRA_OWR_AOA_MEASUREMENT_NTF_PERIOD_SINGLE,
    FIRA_OWR_AOA_MEASUREMENT_NTF_PERIOD_AGGREGATED
};
```

2.3.260.2 Constants

FIRA_OWR_AOA_MEASUREMENT_NTF_PERIOD_SINGLE
notification sent after every received frames.

FIRA_OWR_AOA_MEASUREMENT_NTF_PERIOD_AGGREGATED
notification sent after MIN_FRAMES_PER_RR aggregated frames.

2.3.261 fira_free_frame_report

void **fira_free_frame_report**(struct frame_report *head)
Free the frame report created by diagnostic.

Parameters

- **head** (struct frame_report*) – Head of the frame report.

2.4 PCTT helper API

2.4.1 struct pctt_parameters

struct **pctt_parameters**
PCTT parameters.

2.4.1.1 Definition

```
struct pctt_parameters {
    uint32_t num_packets;
    uint32_t t_gap;
    uint32_t t_start;
    uint32_t t_win;
    uint8_t randomize_psdu;
    uint8_t phr_ranging_bit;
    uint32_t rmarker_tx_start;
    uint32_t rmarker_rx_start;
    uint8_t sts_index_auto_incr;
    uint16_t rssi_outliers;
}
```

2.4.1.2 Members

num_packets
number of packets.

t_gap
Gap between start of one packet to the next in micro seconds.

t_start
Max. time from the start of T_GAP to SFD found state in micro seconds.

t_win
Max. time for which RX is looking for a packet from the start of T_GAP in micro seconds.

randomize_psdu
: 0 no randomization, 1 take first byte of data supplied by cmd is the seed.

phr_ranging_bit
: 0 disable, 1 enable. Configures ranging bit field of PHR.

rmarker_tx_start
: Start time of TX in 1/(128*499.2MHz) units.

rmarker_rx_start
: Start time of RX in 1/(128*499.2MHz) units.

sts_index_auto_incr
: 0x00: STS_INDEX config value is used for all PER Rx/ Periodic TX. 0x01: STS_INDEX value SHALL be incremented for every frame in PER Rx/Periodic TX test.

rssi_outliers
: number of outliers to remove from Rssi values.

2.4.2 struct pctt_test_payload

struct **pctt_test_payload**

PCTT test payload.

2.4.2.1 Definition

```
struct pctt_test_payload {
    uint8_t payload[PCTT_PAYLOAD_MAX_LEN];
    int payload_len;
}
```

2.4.2.2 Members

payload
: PSDU data bytes used by certain cmd tests.

payload_len
: payload length.

2.4.3 struct pctt_session_parameters

struct **pctt_session_parameters**

PCTT session parameters.

2.4.3.1 Definition

```
struct pctt_session_parameters {
    uint8_t device_role;
    uint16_t short_addr;
    uint16_t destination_short_address;
    uint8_t tx_antenna_selection;
    uint8_t rx_antenna_selection;
    uint32_t slot_duration_rstu;
    uint8_t channel_number;
    uint8_t preamble_code_index;
    uint8_t rframe_config;
    uint8_t prf_mode;
    uint8_t preamble_duration;
    uint8_t sfd_id;
    uint8_t number_of_sts_segments;
    uint8_t psdu_data_rate;
    uint8_t phr_data_rate;
    uint8_t mac_fcs_type;
    uint32_t sts_index;
    uint8_t sts_length;
}
```

2.4.3.2 Members

device_role
see enum device_role.

short_addr
Device short address

destination_short_address
Address of controller.

tx_antenna_selection
Selection of TX antenna configuration for this session.

rx_antenna_selection
Selection of RX antenna configuration for this session.

slot_duration_rstu
Duration of a slot in RSTU. (1200RSTU=1ms)

channel_number
Uwb channel for this session.

preamble_code_index
Uwb preamble code index. BPRF (9-24), HPRF (25-32) *[Not supported in QM33 SDK]*

rframe_config
see enum fira_rframe_config.

prf_mode
prf_mode 0x00 = 62.4 MHz PRF. BPRF mode (default) 0x01 = 124.8 MHz PRF. HPRF mode. *[Not supported in QM33 SDK]* 0x02 – 249.6 MHz PRF. HPRF mode with data rate 27.2 and 31.2 Mbps *[Not supported in QM33 SDK]* Values 0x03 to 0xFF = RFU

preamble_duration
preamble_duration 0x00 = 32 symbols *[Not supported in QM33 SDK]* 0x01 = 64 symbols (default) Values 0x02 to 0xFF = RFU

sfd_id
BPRF (O or 2), HPRF (1-4) *[Not supported in QM33 SDK]*.

number_of_sts_segments
number_of_sts_segments 0x01 = 1 STS Segment (default) 0x02 = 2 STS Segments (HPRF only) *[Not supported in QM33 SDK]* 0x03 = 3 STS Segments (HPRF only) *[Not supported in QM33 SDK]* 0x04 = 4 STS Segments (HPRF only) *[Not supported in QM33 SDK]* Values 0x05 to 0xFF = RFU

psdu_data_rate
psdu_data_rate 0x00 - > 6.81Mbps (Default) 0x01 = 7.80 Mbps *[Not supported in QM33 SDK]* 0x02 = 27.2 Mbps *[Not supported in QM33 SDK]* 0x03 = 31.2 Mbps *[Not supported in QM33 SDK]* 0x04 = 850 Kbps *[Not supported in QM33 SDK]* Values 0x00, 0x02, 0x04 map to K=3 and 0x01, 0x03 map to K=7. Values 0x05 to 0xFF = RFU

phr_data_rate
BPRF PHR data rate 0x00 = 850kb/s rate. 0x01 = 6.8Mb/s rate.

mac_fcs_type
mac_fcs_type 0x00 = CRC 16 (default) 0x01 = CRC 32 Values 0x02 to 0xFF = RFU

sts_index
sts_index. default = 0.

sts_length
Number of symbols in a STS segment.

Possible values:

- 0x00: 32 symbols
- 0x01: 64 symbols (default)
- 0x02: 128 symbols
- Values 0x03 to 0xFF: RFU

2.4.4 struct pctt_result_data

struct **pctt_result_data**
PCTT result data.

2.4.4.1 Definition

```
struct pctt_result_data {
    uint8_t status;
    uint32_t attempts;
    uint32_t acq_detect;
    uint32_t acq_reject;
    uint32_t rx_fail;
    uint32_t sync_cir_ready;
    uint32_t sfd_fail;
    uint32_t sfd_found;
    uint32_t phr_dec_error;
    uint32_t phr_bit_error;
```

(continues on next page)

(continued from previous page)

```
    uint32_t psdu_dec_error;
    uint32_t psdu_bit_error;
    uint32_t sts_found;
    uint32_t eof;
    uint32_t rx_done_ts_int;
    uint16_t rx_done_ts_frac;
    uint8_t toa_gap;
    uint8_t phr;
    uint8_t psdu_data[PCTT_PAYLOAD_MAX_LEN];
    uint16_t psdu_data_len;
    uint32_t tx_ts_int;
    uint16_t tx_ts_frac;
    uint32_t rx_ts_int;
    uint16_t rx_ts_frac;
    int16_t noise_value;
    uint32_t measurement;
    int16_t pdoa_azimuth_2pi;
    int16_t pdoa_elevation_2pi;
    uint8_t nb_rssi;
    uint16_t rssis_q8[MAX_RSSI];
    int16_t aoa_azimuth_2pi;
    int16_t aoa_elevation_2pi;
    int32_t cfo_q26;
}
```

2.4.4.2 Members

status
Generic status code.

attempts
Number of RX attempts.

acq_detect
Number of times signal was detected.

acq_reject
Number of times signal was rejected.

rx_fail
Number of times RX did not go beyond ACQ stage.

sync_cir_ready
Number of times sync. CIR ready event was received.

sfd_fail
Number of times RX was stuck at either ACQ detect or sync CIR ready.

sfd_found
Number of times SFD was found.

phr_dec_error
No. of times PHR decode failed.

phr_bit_error
No. of times PHR bits in error.

psdu_dec_error
: No. of times payload decode failed.

psdu_bit_error
: No. of times payload bits in error.

sts_found
: No. of times STS detection was successful.

eof
: No. of times end of frame event was triggered

rx_done_ts_int
: Integer part of timestamp 1/124.8Mhz ticks.

rx_done_ts_frac
: Fractional part of timestamp 1/(128*499.2Mhz) ticks.

toa_gap
: ToA of main path minus ToA of first path in nanoseconds.

phr
: Received PHR (bits 0-12 as per IEEE spec).

psdu_data
: Length of PSDU Data(N) to follow.

psdu_data_len
: Length of psdu_data.

tx_ts_int
: Integer part of timestamp in 1/124.8 us resolution.

tx_ts_frac
: Fractional part of timestamp in 1/124.8/512 us resolution.

rx_ts_int
: Integer part of timestamp in 1/124.8 us resolution.

rx_ts_frac
: Fractional part of timestamp in 1/124.8/512 us resolution.

noise_value
: General noise value in dB, allowing to compute SNR.

measurement
: For TEST_SS_TWR_NTF. Contains Tround time of Initiator or Treply time of Responder depending on DEVICE_ROLE option. This is expressed in 1/(128 * 499.2Mhz) ticks.

pdoa_azimuth_2pi
: Estimation of reception phase difference in the azimuth.

pdoa_elevation_2pi
: Estimation of reception phase difference in the elevation.

nb_rssi
: The number of RSSIs in the array *rssis_q8*.

rssis_q8
: Calculated RSSIs (Received Signal Strength Indicator), encoded as Q8.8.

aoa_azimuth_2pi
: Estimation of reception angle in the azimuth.

aoa_elevation_2pi
: Estimation of reception angle in the elevation.

cfo_q26
Carrier Frequency Offset, encoded as signed q8.26.

2.4.5 typedef pctt_helper_notification_cb_t

void **pctt_helper_notification_cb_t**(void *user_data, uint8_t cmd_id, const struct *pctt_result_data* *results)

Notification callback type.

Parameters

- **user_data** (void*) – User data pointer given to pctt_helper_open.
- **cmd_id** (uint8_t) – The type of cmd corresponding to the results.
- **results** (const struct *pctt_result_data**) – pctt results.

2.4.5.1 Return

nothing

2.4.6 pctt_helper_open

enum qerr **pctt_helper_open**(struct pctt_context *context, struct *uwbmac_context* *uwbmac_context, *pctt_helper_notification_cb_t* notification_cb, const char *scheduler, int region_id, void *user_data)

Initialize the internal resources of the helper.

Parameters

- **context** (struct pctt_context*) – Context to initialize.
- **uwbmac_context** (struct *uwbmac_context**) – UWB MAC context.
- **notification_cb** (*pctt_helper_notification_cb_t*) – Callback to call when a notification is available.
- **scheduler** (const char*) – In which scheduler the region will be
- **region_id** (int) – Which id the region will have in the scheduler
- **user_data** (void*) – User data pointer to give back in callbacks.

2.4.6.1 NOTE

This function must be called first. `pctt_helper_close` must be called at the end of the application to ensure resources are freed.

2.4.6.2 Return

QERR_SUCCESS or error.

2.4.7 pctt_helper_set_scheduler

enum qerr **pctt_helper_set_scheduler**(struct pctt_context *context)

Set the scheduler and the region.

Parameters

- **context** (struct pctt_context*) – Context of this helper.

2.4.7.1 NOTE

This function must be called while the UWB MAC is stopped.

2.4.7.2 Return

QERR_SUCCESS or error.

2.4.8 pctt_helper_close

void **pctt_helper_close**(struct pctt_context *context)

Free all internal resources of the helper.

Parameters

- **context** (struct pctt_context*) – Context to release.

2.4.9 pctt_helper_session_init

int **pctt_helper_session_init**(struct pctt_context *context)

Initialize a pctt session.

Parameters

- **context** (struct pctt_context*) – Context of this helper.

2.4.9.1 Description

This function must be called first to create and initialize the pctt session.

2.4.9.2 Return

0 or error.

2.4.10 pctt_helper_session_start

int **pctt_helper_session_start**(struct pctt_context *context, uint8_t cmd_id, const struct *pctt_parameters* *params)

Start a pctt session.

Parameters

- **context** (struct pctt_context*) – Context of this helper.
- **cmd_id** (uint8_t) – The cmd being executed.
- **params** (const struct *pctt_parameters**) – specific pctt_parameters of the test.

2.4.10.1 Description

This function must be called after pctt session was initialized.

2.4.10.2 Return

0 or error.

2.4.11 pctt_helper_set_test_payload

int **pctt_helper_set_test_payload**(struct pctt_context *context, const struct *pctt_test_payload* *test_payload)

Set payload.

Parameters

- **context** (struct pctt_context*) – Context of this helper.
- **test_payload** (const struct *pctt_test_payload**) – data payload for the test.

2.4.11.1 Description

This function must be called after pctt session was initialized.

2.4.11.2 Return

0 or error.

2.4.12 pctt_helper_session_deinit

int **pctt_helper_session_deinit**(struct pctt_context *context)

Deinitialize a pctt session.

Parameters

- **context** (struct pctt_context*) – Context of this helper.

2.4.12.1 Description

This function is called to free all memory allocated by the session. This function must be called when the session is stopped.

2.4.12.2 Return

0 or error.

2.4.13 pctt_helper_session_get_state

int **pctt_helper_session_get_state**(struct pctt_context *context, int *state)

Request to get a state.

Parameters

- **context** (struct pctt_context*) – Context of this helper.
- **state** (int*) – The result state of the session.

2.4.13.1 Return

0 or error.

2.4.14 pctt_helper_session_get_params

int **pctt_helper_session_get_params**(struct pctt_context *context, struct *pctt_session_parameters* *params)

Request pctt parameters.

Parameters

- **context** (struct pctt_context*) – Context of this helper.
- **params** (struct *pctt_session_parameters**) – the output parameters.

2.4.14.1 Return

0 or error.

2.4.15 pctt_helper_session_set_params

int **pctt_helper_session_set_params**(struct pctt_context *context, struct *pctt_session_parameters* *params)

Set pctt session parameters.

Parameters

- **context** (struct pctt_context*) – Context of this helper.
- **params** (struct *pctt_session_parameters**) – the session parameters to set.

2.4.15.1 Return

0 or error.

2.4.16 pctt_helper_tx_cw

enum qerr **pctt_helper_tx_cw**(struct pctt_context *context, struct *pctt_session_parameters* *params, bool start)

Start/stop continuous wave test.

Parameters

- **context** (struct pctt_context*) – Context of this helper.
- **params** (struct *pctt_session_parameters**) – Specific pctt_parameters of the test, used to set channel_number and TX antenna to transit from.
- **start** (bool) – Whether to start or not.

2.4.16.1 Return

QERR_SUCCESS or error.

2.4.17 pctt_helper_pll_lock

enum qerr **pctt_helper_pll_lock**(struct pctt_context *context, struct *pctt_session_parameters* *params, uint8_t *errcode, uint32_t *pll_status)

Run PLL Lock test.

Parameters

- **context** (struct pctt_context*) – Context of this helper.
- **params** (struct *pctt_session_parameters**) – Specific pctt_parameters, used to set channel_number.
- **errcode** (uint8_t*) – Returned error code.
- **pll_status** (uint32_t*) – Returned PLL status in case of test success.

2.4.17.1 Return

QERR_SUCCESS or error.

2.4.18 enum pctt_device_role

enum **pctt_device_role**

[NOT IMPLEMENTED] Role played by a device.

2.4.18.1 Definition

```
enum pctt_device_role {
    PCTT_DEVICE_ROLE_RESPONDER,
    PCTT_DEVICE_ROLE_INITIATOR
};
```

2.4.18.2 Constants

PCTT_DEVICE_ROLE_RESPONDER
The device acts as a responder.

PCTT_DEVICE_ROLE_INITIATOR
The device acts as an initiator.

2.4.18.3 Description

Current implementation does not support decorrelation between the device's role and the device's type. The controller is always the initiator and the controlee is always the responder.

This enum is not used in the current implementation.

2.4.19 enum pctt_rframe_config

enum **pctt_rframe_config**

Rframe configuration used to transmit/receive ranging messages.

2.4.19.1 Definition

```
enum pctt_rframe_config {
    PCTT_RFRAME_CONFIG_SP0,
    PCTT_RFRAME_CONFIG_SP1,
    PCTT_RFRAME_CONFIG_SP2,
    PCTT_RFRAME_CONFIG_SP3
};
```

2.4.19.2 Constants

PCTT_RFRAME_CONFIG_SP0
Use SP0 mode.

PCTT_RFRAME_CONFIG_SP1
Use SP1 mode.

PCTT_RFRAME_CONFIG_SP2
RFU

PCTT_RFRAME_CONFIG_SP3
Use SP3 mode.

2.4.20 enum pctt_prf_mode

enum **pctt_prf_mode**
Pulse Repetition Frequency mode.

2.4.20.1 Definition

```
enum pctt_prf_mode {
    PCTT_PRF_MODE_BPRF,
    PCTT_PRF_MODE_HPRF,
    PCTT_PRF_MODE_HPRF_HIGH_RATE
};
```

2.4.20.2 Constants

PCTT_PRF_MODE_BPRF
Base Pulse Repetition Frequency.

PCTT_PRF_MODE_HPRF
[Not supported in QM33 SDK] Higher Pulse Repetition Frequency.

PCTT_PRF_MODE_HPRF_HIGH_RATE
[Not supported in QM33 SDK] Higher Pulse Repetition Frequency allowing higher data rates (27M2 and 31M2).

2.4.20.3 Description

This enum is not used in the current implementation.

2.4.21 enum pctt_preamble_duration

enum **pctt_preamble_duration**
Duration of preamble in symbols.

2.4.21.1 Definition

```
enum pctt_preamble_duration {
    PCTT_PREAMBLE_DURATION_32,
    PCTT_PREAMBLE_DURATION_64
};
```

2.4.21.2 Constants

PCTT_PREAMBLE_DURATION_32
: *[Not supported in QM33 SDK]* 32 symbols duration.

PCTT_PREAMBLE_DURATION_64
: 64 symbols duration.

2.4.22 enum pctt_sfd_id

enum **pctt_sfd_id**
: Start-of-frame delimiter.

2.4.22.1 Definition

```
enum pctt_sfd_id {
    PCTT_SFD_ID_0,
    PCTT_SFD_ID_1,
    PCTT_SFD_ID_2,
    PCTT_SFD_ID_3,
    PCTT_SFD_ID_4
};
```

2.4.22.2 Constants

PCTT_SFD_ID_0
: Delimiter is [0 +1 0 –1 +1 0 0 –1]

PCTT_SFD_ID_1
: *[Not supported in QM33 SDK]* Delimiter is [–1 –1 +1 –1]

PCTT_SFD_ID_2
: Delimiter is [–1 –1 –1 +1 –1 –1 +1 –1]

PCTT_SFD_ID_3
: *[Not supported in QM33 SDK]* Delimiter is [–1 –1 –1 –1 –1 +1 +1 –1 –1 +1 –1 +1 –1 –1 +1 –1]

PCTT_SFD_ID_4
: *[Not supported in QM33 SDK]* Delimiter is [–1 –1 –1 –1 –1 –1 –1 +1 –1 –1 +1 –1 –1 +1 –1 +1 –1 +1 –1 –1 –1 +1 +1 –1 –1 –1 +1 –1 +1 +1 –1 –1]

2.4.23 enum pctt_number_of_sts_segments

enum **pctt_number_of_sts_segments**

Number of STS segments.

2.4.23.1 Definition

```
enum pctt_number_of_sts_segments {
    PCTT_NUMBER_OF_STS_SEGMENTS_NONE,
    PCTT_NUMBER_OF_STS_SEGMENTS_1_SEGMENT,
    PCTT_NUMBER_OF_STS_SEGMENTS_2_SEGMENTS,
    PCTT_NUMBER_OF_STS_SEGMENTS_3_SEGMENTS,
    PCTT_NUMBER_OF_STS_SEGMENTS_4_SEGMENTS
};
```

2.4.23.2 Constants

PCTT_NUMBER_OF_STS_SEGMENTS_NONE
No STS Segment (Rframe config SP0).

PCTT_NUMBER_OF_STS_SEGMENTS_1_SEGMENT
1 STS Segment.

PCTT_NUMBER_OF_STS_SEGMENTS_2_SEGMENTS
[Not supported in QM33 SDK] 2 STS Segments.

PCTT_NUMBER_OF_STS_SEGMENTS_3_SEGMENTS
[Not supported in QM33 SDK] 3 STS Segments.

PCTT_NUMBER_OF_STS_SEGMENTS_4_SEGMENTS
[Not supported in QM33 SDK] 4 STS Segments.

2.4.24 enum pctt_psdu_data_rate

enum **pctt_psdu_data_rate**

Data rate used to exchange PSDUs.

2.4.24.1 Definition

```
enum pctt_psdu_data_rate {
    PCTT_PSDU_DATA_RATE_6M81,
    PCTT_PSDU_DATA_RATE_7M80,
    PCTT_PSDU_DATA_RATE_27M2,
    PCTT_PSDU_DATA_RATE_31M2
};
```

2.4.24.2 Constants

PCTT_PSDU_DATA_RATE_6M81
6.8Mb/s rate.

PCTT_PSDU_DATA_RATE_7M80
[Not supported in QM33 SDK] 7.8Mb/s rate.

PCTT_PSDU_DATA_RATE_27M2
[Not supported in QM33 SDK] 27.2Mb/s rate.

PCTT_PSDU_DATA_RATE_31M2
[Not supported in QM33 SDK] 31.2Mb/s rate.

2.4.25 enum pctt_phr_data_rate

enum **pctt_phr_data_rate**
Data rate used to exchange PHR.

2.4.25.1 Definition

```
enum pctt_phr_data_rate {
    PCTT_PHR_DATA_RATE_850K,
    PCTT_PHR_DATA_RATE_6M81
};
```

2.4.25.2 Constants

PCTT_PHR_DATA_RATE_850K
850kb/s rate.

PCTT_PHR_DATA_RATE_6M81
6.8Mb/s rate.

2.4.25.3 Description

This enum is not used in the current implementation.

2.4.26 enum pctt_status_ranging

enum **pctt_status_ranging**
Ranging status: success or failure reason.

2.4.26.1 Definition

```
enum pctt_status_ranging {
    PCTT_STATUS_RANGING_INTERNAL_ERROR,
    PCTT_STATUS_RANGING_SUCCESS,
    PCTT_STATUS_RANGING_TX_FAILED,
    PCTT_STATUS_RANGING_RX_TIMEOUT,
    PCTT_STATUS_RANGING_RX_PHY_DEC_FAILED,
    PCTT_STATUS_RANGING_RX_PHY_TOA_FAILED,
    PCTT_STATUS_RANGING_RX_PHY_STS_FAILED,
    PCTT_STATUS_RANGING_RX_MAC_DEC_FAILED,
    PCTT_STATUS_RANGING_RX_MAC_IE_DEC_FAILED,
    PCTT_STATUS_RANGING_RX_MAC_IE_MISSING
};
```

2.4.26.2 Constants

PCTT_STATUS_RANGING_INTERNAL_ERROR
Implementation specific error.

PCTT_STATUS_RANGING_SUCCESS
Ranging info are valid.

PCTT_STATUS_RANGING_TX_FAILED
Failed to transmit UWB packet.

PCTT_STATUS_RANGING_RX_TIMEOUT
No UWB packet detected by the receiver.

PCTT_STATUS_RANGING_RX_PHY_DEC_FAILED
UWB packet channel decoding error.

PCTT_STATUS_RANGING_RX_PHY_TOA_FAILED
Failed to detect time of arrival of the UWB packet from CIR samples.

PCTT_STATUS_RANGING_RX_PHY_STS_FAILED
UWB packet STS segment mismatch.

PCTT_STATUS_RANGING_RX_MAC_DEC_FAILED
MAC CRC or syntax error.

PCTT_STATUS_RANGING_RX_MAC_IE_DEC_FAILED
IE syntax error.

PCTT_STATUS_RANGING_RX_MAC_IE_MISSING
Expected IE missing in the packet.

2.4.27 enum pctt_session_state

enum **pctt_session_state**

Session state.

2.4.27.1 Definition

```
enum pctt_session_state {
    PCTT_SESSION_STATE_INIT,
    PCTT_SESSION_STATE_DEINIT,
    PCTT_SESSION_STATE_ACTIVE,
    PCTT_SESSION_STATE_IDLE
};
```

2.4.27.2 Constants

PCTT_SESSION_STATE_INIT
Initial state, session is not ready yet.

PCTT_SESSION_STATE_DEINIT
Session does not exist.

PCTT_SESSION_STATE_ACTIVE
Session is currently active.

PCTT_SESSION_STATE_IDLE
Session is ready to start, but not currently active.

2.4.28 enum pctt_sts_length

enum **pctt_sts_length**

Number of symbols in a STS segment.

2.4.28.1 Definition

```
enum pctt_sts_length {
    PCTT_STS_LENGTH_32,
    PCTT_STS_LENGTH_64,
    PCTT_STS_LENGTH_128
};
```

2.4.28.2 Constants

PCTT_STS_LENGTH_32
The STS length is 32 symbols.

PCTT_STS_LENGTH_64
The STS length is 64 symbols.

PCTT_STS_LENGTH_128
The STS length is 128 symbols.

Index